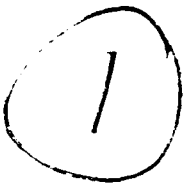
AD-A244 928



SOFTWARE DESIGN DOCUMENT
MCC CSCI (1)

Volume 1 of 2 Sections 1.0 - 2.18

DTIC
ELECTE
JAN 09 1992
S D
D

June, 1991

Prepared by:

BBN Systems and Technologies,
A Division of Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, MA 02138
(617) 873-3000 FAX: (617) 873-4315

Prepared for:

Defense Advanced Research Projects Agency (DARPA)
Information and Science Technology Office
1400 Wilson Blvd., Arlington, VA 22209-2308
(202) 694-8232, AUTOVON 224-8232

Program Manager for Training Devices (PM TRADE)
12350 Research Parkway
Orlando, FL 32826-3276
(407) 380-4518

92-00264

92 1 6 070

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

REPORT DOCUMENTATION PAGE

Form Approved
OPM No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources gathering and maintaining the data needed, and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Information and Regulatory Affairs, Office of Management and Budget, Washington, DC 20503.

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1991	Software Design Document

4. TITLE AND SUBTITLE

Software Design Document MCC CSCI (1)

5. FUNDING NUMBERS

Contract Numbers:
MDA972-89-C-0060
MDA972-89-C-0061

6. AUTHOR(S)

Author not specified.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Bolt Beranek and Newman, Inc. (BBN)
Systems and Technologies; Advanced Simulation
10 Moulton Street
Cambridge, MA 02138

8. PERFORMING ORGANIZATION REPORT NUMBER

Advanced Simulation #:
9104

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Projects Agency (DARPA)
3701 North Fairfax Drive
Arlington, VA 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

DARPA Report Number:
None.

11. SUPPLEMENTARY NOTES

None

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution Statement A: Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

Distribution Code:
A

13. ABSTRACT *(Maximum 200 words)*

A Simulation Network (SIMNET) project Software Design Document that describes the Management, Command, and Control (MCC) Computer Software Configuration Item (CSCI number 1) of the SIMNET hardware and software training system for vehicle crew training and operational training.

14. SUBJECT TERMS

SIMNET Software Design Document for the MCC CSCI (CSCI 1).

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Same as report.

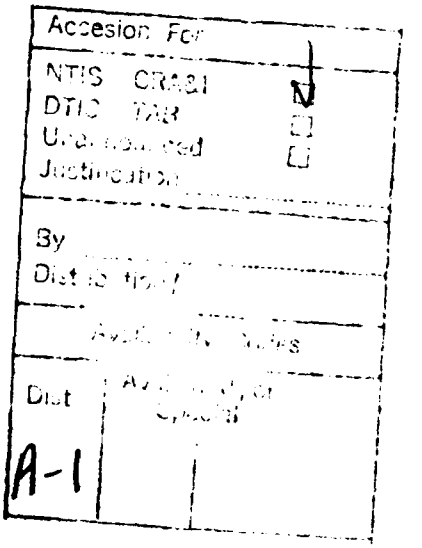
SOFTWARE DESIGN DOCUMENT
MCC CSCI (1)

Volume 1 of 2 Sections 1.0 - 2.18

June, 1991

Prepared by:

BBN Systems and Technologies,
A Division of Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, MA 02138
(617) 873-3000 FAX: (617) 873-4315



Prepared for:

Defense Advanced Research Projects Agency (DARPA)
Information and Science Technology Office
1400 Wilson Blvd., Arlington, VA 22209-2308
(202) 694-8232, AUTOVON 224-8232

Program Manager for Training Devices (PM TRADE)
12350 Research Parkway
Orlando, FL 32826-3276
(407) 380-4518

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

Table of Contents

1 INTRODUCTION: MCC CSCI DESCRIPTION 1
1.1 BACKGROUND 1
1.2 EXTERNAL INTERFACES 1
1.3 INTERNAL STRUCTURE 3
1.4 CONFIGURATION AND CONFIGURATION MANAGEMENT 7
1.5 TERMINOLOGY AND DOCUMENTATION 7
2 CSC DESCRIPTIONS 8
2.1 THE MOTHER PROCESS 8
2.1.1 Initializing MCC Software 9
2.1.1.1 data.c 9
2.1.1.2 mother.c 10
2.1.1.2.1 ProcessMessage 10
2.1.1.2.2 EmplaceRequest 11
2.1.1.2.3 BreachRequest 11
2.1.1.2.4 ResupplyOffer 12
2.1.1.2.5 PDU_Dispatch_NOP 13
2.1.1.2.6 Init_PDU_Dispatch 14
2.1.1.2.7 Check_Net 14
2.1.1.2.8 Init_MCC_Processing 15
2.1.1.2.9 main 16
2.1.1.3 param.c 16
2.1.1.3.1 Battalion 17
2.1.1.3.2 Bridge 17
2.1.1.3.3 Console 18
2.1.1.3.3 Ex_Log 18
2.1.1.3.4 Exercise 19
2.1.1.3.5 Password 19
2.1.1.3.6 Smoke 20
2.1.1.3.7 Terrain 20
2.1.1.3.8 Veh_Log 21
2.1.1.3.9 Vehicle 21
2.1.1.3.10 VList 22
2.1.1.3.11 VehHashTableSize 23
2.1.1.3.12 StatusHashSize 23
2.1.1.3.13 VehicleMarkingR 23
2.1.1.3.14 CurBattleScheme 24
2.1.1.3.15 DefineBattleScheme 24

2.1.1.3.16 MineScanInterval 25
2.1.1.3.17 MarkerSpacing 25
2.1.1.3.18 MarkerInterval 26
2.1.1.3.19 ProcessParameters 26
2.1.1.3.20 ScanString 27
2.1.1.4 process.c 27
2.1.1.4.1 InitProcessTable 28
2.1.1.5 vehicle.c 28
2.1.1.5.1 AllocShMem 28
2.1.1.5.2 InitVehicleTables 29
2.1.1.5.3 VehicleIDToSim 29
2.1.1.5.4 GetVehicleLocation 30
2.1.1.5.5 IsGroundVehicle 31
2.1.1.5.6 IsCombatVehicle 31
2.1.2 Modeling Computer-Controlled Vehicles (CCVs) 32
2.1.2.1 cas.c 32
2.1.2.1.1 PlaceBombs 32
2.1.2.1.2 GuideBomber 33
2.1.2.1.3 ConsiderVehicle 34
2.1.2.2 popden.c 35
2.1.2.2.1 InitPopDenMap 35
2.1.2.2.2 PDMInsert 36
2.1.2.2.3 PDMDelete 36
2.1.2.2.4 NearbyCCVs 37
2.1.2.3 ccv_model.c 38
2.1.2.3.1 InitCCV 38
2.1.2.3.2 BroadcastCCVBatch 39
2.1.2.3.3 BroadcastAppearance 39
2.1.2.3.4 BroadcastDeactivate 40
2.1.2.3.5 FireShell 40
2.1.2.4 fire.c 41
2.1.2.4.1 InitIndirectFire 41
2.1.2.4.2 FireVolley 42
2.1.2.4.3 CheckVolleyTable 42
2.1.2.4.4 PlaceIndirectFire 43
2.1.2.4.5 SendIndirectFirePDU 44
2.1.2.4.6 IndirectFireDamage 45
2.1.2.5 placement.c 45
2.1.2.5.1 FindAnOpenSpot 45
2.1.2.5.2 NearbyVehicle 47
2.1.2.5.3 NearestCombatVehicle 48
2.1.2.6 ccv_change.c 49
2.1.2.6.1 PlaceCCV 49

2.1.2.6.2 VehicleIsGunneryTarget 50
2.1.2.6.3 RemoveCCV 50
2.1.2.6.4 AttackCCV 51
2.1.3 Positioning Vehicles 51
2.1.3.1 soil.c 52
2.1.3.1.1 SoilOkay 52
2.1.4 Activating Combat Vehicle Simulators 52
2.1.4.1 actloc.c 53
2.1.4.1.1 Act_List_Hash 53
2.1.4.1.2 Act_List_Alloc 53
2.1.4.1.3 Act_List_Init 54
2.1.4.1.4 Act_List_Add 54
2.1.4.1.5 Act_List_Delete 55
2.1.4.1.6 Act_List_Loc_Exists 55
2.1.4.2 sim.c 56
2.1.4.2.1 UpdateAppearance 56
2.1.4.2.2 ActivateForExerciseStart 57
2.1.4.2.3 ActivateForReconstitution 57
2.1.4.2.4 ActivateForTowingArrival 58
2.1.4.2.5 Build_ActivatePDU 59
2.1.4.2.6 ActivateReplyReceived 60
2.1.4.2.7 ActivateReplyTimedOut 60
2.1.4.2.8 ActivateVehicle 61
2.1.4.2.9 DeactivateReplyReceived 63
2.1.4.2.10 DeactivateReplyTimedOut 63
2.1.4.2.11 DeactivateVehicle 64
2.1.4.2.12 CheckSimulatorActivity 65
2.1.5 Recording Statistics 66
2.1.5.1 shutdown.c 66
2.1.5.1.1 ResetMCC 66
2.1.5.1.2 DeactivateAllVehicles 66
2.1.5.1.3 KillServiceProcesses 67
2.1.5.1.4 LogExerciseEnd 67
2.1.5.1.5 TimeString 68
2.1.6 Listening to the SIMNET Local Area Network 69
2.1.6.1 maint.c 69
2.1.6.1.1 RepairReplyReceived 69
2.1.6.1.2 RepairReplyTimedOut 69
2.1.6.1.3 Repair_Request 70
2.1.6.2 protocol.c 71
2.1.6.2.1 SaveVehicleStatus 71
2.1.6.2.2 ProcessDatColDatagram 72
2.1.6.2.3 ProcessDataColTransaction 73

2.1.6.2.4 ProcessMgmtDatagram 73
2.1.6.2.5 ProcessMgmtTransaction 74
2.1.6.2.6 ProcessSimTransaction 74
2.1.6.2.7 Remap_Ammo 75
2.1.6.2.8 Remap_Fuel 76
2.1.6.2.9 ProcessSimDatagram 76
2.1.6.2.10 CountRoundsFired 77
2.1.7 Monitoring of the MCC System 78
2.1.7.1 status.c 78
2.1.7.1.1 ReportMCCSimulationStatus 78
2.1.7.1.2 same_unit 79
2.1.7.1.3 included_unit 80
2.1.7.1.4 including_unit 80
2.1.7.1.5 IgnoreStatusQuery 81
2.1.7.1.6 Build_Exercise_Status 82
2.1.7.1.7 Build_Simulation_Status 83
2.1.7.1.8 Build_Vehicle_Status 83
2.1.7.1.9 Do_Generic_StatusQuery 84
2.1.7.1.10 ProcessStatusQueryDatagram 85
2.1.7.1.11 ProcessStatusQueryTransaction 86
2.1.7.2 vtimeout.c 87
2.1.7.2.1 VTimeout_Appearance_Timeout 87
2.1.8 Minefield Simulation 88
2.1.8.1 geometry.c 88
2.1.8.1.1 line_crosses_rect 88
2.1.8.1.2 intersect_rect 89
2.1.8.1.3 init_rect 90
2.1.8.1.4 inflate_rect 90
2.1.8.1.5 make_polygon 90
2.1.8.1.6 point_in_polygon 91
2.1.8.2 marker.c 92
2.1.8.2.1 Marker_Emplace_Minefield 93
2.1.8.2.2 Marker_Add_Line 94
2.1.8.2.3 Marker_Broadcast_Markers 95
2.1.8.2.4 Marker_Get_Next_Marker 95
2.1.8.2.5 Marker_Point_In_Terrain 96
2.1.8.3 minefield.c 96
2.1.8.3.1 Current_Minefield_Count 96
2.1.8.3.2 Alloc_Minefield 97
2.1.8.3.3 Free_Minefield 97
2.1.8.3.4 Mines_Max_Minefields 98
2.1.8.3.5 Mines_Max_Vertices 98
2.1.8.3.6 Emplace_Generic 99

2.1.8.3.7 Mines_Emplace_Minefield 99
2.1.8.3.8 Mines_Breach_Lane 100
2.1.8.3.9 Mines_Emplace_Breach 101
2.1.8.3.10 Mines_Process_Breach_Datagram 102
2.1.8.3.11 Mines_Get_Info 102
2.1.8.3.12 Blow_Vehicle 103
2.1.8.3.13 Mines_Point_In_Minefield 104
2.1.8.3.14 Crossing_Minefield 105
2.1.8.3.15 Mines_Check_Minefields 106
2.1.8.3.16 Make_Breach_Status 106
2.1.8.3.17 Make_Minefield_Status 107
2.1.8.3.18 Mines_Send_Minefield_Status 107
2.2 THE SCC (SIMNET CONTROL CONSOLE) PROCESS 109
2.2.1 Initialization 111
2.2.1.1 cas.c 112
2.2.1.1.1 ProcessCASRequest 112
2.2.1.2 css.c 112
2.2.1.2.1 ProcessTruckInitRequest 113
2.2.1.2.2 InitializeTruck 113
2.2.1.2.3 ReconstituteTruck 114
2.2.1.2.4 ProcessUMCPDisplaceRequest 114
2.2.1.2.5 ProcessDepotRequest 115
2.2.1.2.6 ProcessCSSRequest 115
2.2.1.2.7 ProcessTruckQueryRequest 116
2.2.1.3 fse.c 117
2.2.1.3.1 ProcessFSERequest 117
2.2.1.3.2 ProcessArtyQueryRequest 118
2.2.1.3.3 ProcessArtyReconstRequest 118
2.2.1.3.4 ProcessArtyDispatchRequest 119
2.2.1.3.5 ProcessArtyArriveRequest 120
2.2.1.4 main.c 120
2.2.1.4.1 main 120
2.2.1.4.2 ProcessMessage 121
2.2.1.4.3 RequestArrived 122
2.2.1.4.4 ResetClock 123
2.2.1.4.5 BroadcastToChildren 124
2.2.1.4.6 ProcessPermitOptionsRequest 124
2.2.1.5 terrain.c 125
2.2.1.5.1 ProcessTerrainRequest 125
2.2.1.5.2 ProcessMapSheetsRequest 126
2.2.1.5.3 ProcessExerciseRequest 126
2.2.1.6 cew.c 127
2.2.1.6.1 ProcessCEWInitRequest 127

2.2.1.6.2 ProcessCEWQueryRequest....128
2.2.1.6.3 ProcessCEWStartRequest....129
2.2.1.6.4 GetSCCVehicleIndex....129
2.2.1.6.5 ProcessCEWReconstRequest....130
2.2.2 Placing Combat Vehicles....130
2.2.2.1 sim.c....130
2.2.2.1.1 ProcessSimAllocRequest....130
2.2.2.1.2 SimPlacedLocally....131
2.2.2.1.3 SimAllocedRemotely....132
2.2.2.1.4 SimPlacedRemotely....132
2.2.2.1.5 ReportSimProblem....133
2.2.3 Placing Static Vehicles....133
2.2.3.1 ccv.c....134
2.2.3.1.1 ProcessTOCRequest....134
2.2.3.1.2 ProcessTOCDispatchRequest....135
2.2.3.1.3 ProcessALOCRequest....135
2.2.3.1.4 ProcessALOCDispatchRequest....136
2.2.3.2 target.c....136
2.2.3.2.1 ProcessTargetSetRequest....137
2.2.3.2.2 TargetKilled....137
2.2.4 Communication with World....138
2.2.4.1 restart.c....138
2.2.4.1.1 RestartConsole....138
2.2.4.1.2 Respawn....139
2.2.4.2 stop.c....139
2.2.4.2.1 ProcessStopRequest....140
2.2.4.2.2 ProcessExitMessage....140
2.2.4.2.3 ExerciseStopped....140
2.2.4.2.4 ShutdownChildren....141
2.3 THE PLACE (PLACEMENT) PROCESS....142
2.3.1 Initialization Software....144
2.3.1.1 main.c....144
2.3.1.1.1 main....144
2.3.1.1.2 ProcessMessage....144
2.3.1.1.3 RequestArrived....145
2.3.1.1.4 ProcessStartRequest....146
2.3.1.1.5 ResetClock....146
2.3.2 Placement of Combat Vehicle Simulators....147
2.3.2.1 sim.c....147
2.3.2.1.1 SimAllocedRemotely....147
2.3.2.1.2 SimPlacedLocally....148
2.3.2.1.3 SimPlacedRemotely....148

2.4 THE ADMIN PROCESS....150
2.4.1 Initialization and Communication....152
2.4.1.1 main.c....152
2.4.1.1.1 main....152
2.4.1.1.2 ProcessMessage....153
2.4.1.1.3 ResetClock....153
2.4.1.1.4 RequestArrived....154
2.4.1.1.5 SendForceRequest....154
2.4.1.1.6 SendDepotRequest....155
2.4.1.1.7 InitializeAllTrucks....155
2.4.1.1.8 LookupTruckNumber....156
2.4.1.1.9 LookupCCVNumber....156
2.4.2 Truck Dispatching....157
2.4.2.1 displace.c....157
2.4.2.1.1 InitializeTruck....157
2.4.2.1.2 ReconstituteTruck....158
2.4.2.1.3 TruckArrived....158
2.4.2.1.4 TruckDispatched....159
2.4.2.1.5 TruckKilled....160
2.4.2.1.6 RestoreTruck....160
2.4.2.1.7 TurnFleetIntoCapabilities....161
2.4.3 Servicing....161
2.4.3.1 service.c....162
2.4.3.1.1 InitializeResupply....162
2.4.3.1.2 ServiceRequest....162
2.4.3.1.3 VehicleResupplied....163
2.4.3.1.4 CancelResupply....164
2.4.3.1.5 WithdrawOffer....165
2.4.3.1.6 CancelOffers....166
2.4.3.1.7 RemoveClient....166
2.4.3.1.8 ServiceTimeout....167
2.4.3.1.9 ReadyTimeout....167
2.5 THE MAINT (MAINTENANCE) PROCESS....169
2.5.1 Initialization and Communication....171
2.5.1.1 main.c....171
2.5.1.1.1 main....171
2.5.1.1.2 ProcessMessage....171
2.5.1.1.3 ResetClock....172
2.5.1.1.4 RequestArrived....173
2.5.1.1.5 SendForceRequest....173
2.5.1.1.6 SendDepotRequest....174
2.5.1.1.7 InitializeAllTrucks....174

2.5.1.1.8 LookupTruckNumber....175
2.5.1.1.9 LookupCCVNumber....175
2.5.2 Truck Dispatching....176
2.5.2.1 displace.c....176
2.5.2.1.1 InitializeTruck....176
2.5.2.1.2 ReconstituteTruck....177
2.5.2.1.3 TruckArrived....177
2.5.2.1.4 TruckDispatched....178
2.5.2.1.5 TruckKilled....179
2.5.2.1.6 RestoreTruck....179
2.5.2.1.7 TurnFleetIntoCapabilities....180
2.5.3 Servicing....181
2.5.3.1 service.c....181
2.5.3.1.1 ServiceRequest....181
2.5.3.1.2 PerformRepair....182
2.5.3.1.3 ReadyTimeout....183
2.6 THE FSE (FIRE SUPPORT ELEMENT) PROCESS....184
2.6.1 Initialization and Communication....186
2.6.1.1 main.c....186
2.6.1.1.1 main....186
2.6.1.1.2 ProcessMessage....186
2.6.1.1.3 RequestArrived....187
2.6.1.1.4 InitializeBattery....188
2.6.1.1.5 ReconstituteBattery....189
2.6.1.1.6 TubeKilled....189
2.6.1.1.7 ResetClock....190
2.6.1.2 data.c....190
2.6.2 Positioning of Howitzers and Mortars....190
2.6.2.1 position.c....191
2.6.2.1.1 IsMortar....191
2.6.2.1.2 PlaceHalf....191
2.6.2.1.3 LocalDispatchRequest....192
2.6.2.1.4 LocalArriveRequest....193
2.6.2.1.5 RemoteDispatch....193
2.6.2.1.6 RemoteArrive....194
2.6.3 Delivery of Fire....194
2.6.3.1 fire.c....195
2.6.3.1.1 FireRequest....195
2.6.3.1.2 AdjustFire....195
2.6.3.1.3 FireForEffect....196
2.6.3.1.4 FPFRequest....196
2.6.3.1.5 MapAmmoCode....197
2.6.3.1.6 TimeOfFlight....198

2.6.3.1.7 PlaceParallelSheaf....198
2.6.3.1.8 DistributeBursts....199
2.6.3.1.9 RandomDistance....200
2.6.3.1.10 RandomDelay....200
2.6.3.1.11 UpdateSharedMemory....200
2.7 THE CAS (CLOSE AIR SUPPORT) PROCESS....202
2.7.1 Initialization and Communication....203
2.7.1.1 cas.c....203
2.7.1.1.1 main....203
2.7.1.1.2 ProcessMessage....203
2.7.1.1.3 RequestArrived....204
2.7.1.1.4 AllotSorties....205
2.7.1.1.5 BombRequest....205
2.7.1.1.6 DropBombs....206
2.7.1.1.7 ResetClock....206
2.8 THE CEW (COMBAT ENGINEERING WORKSTATION) PROCESS....207
2.8.1 Initialization and Communication....207
2.8.1.1 main.c....207
2.8.1.1.1 main....207
2.8.1.1.2 ProcessMessage....208
2.8.1.1.3 ResetClock....209
2.8.1.1.4 RequestArrived....209
2.8.1.1.5 ProcessBoundsRequest....210
2.8.1.1.6 ProcessEmplaceRequest....210
2.8.1.1.7 ProcessBreachRequest....211
2.8.1.1.8 ProcessDispatchRequest....211
2.8.1.1.9 ProcessArrivalRequest....212
2.8.1.1.10 ProcessSupplyRequest....213
2.8.1.2 CEWHost.h....213
2.8.2 CEW Asset Management....214
2.8.2.1 asset.h....214
2.8.2.2 asset.c....214
2.8.2.2.1 LookupAssetNumber....214
2.8.2.2.2 InitializeAssets....215
2.8.2.2.3 InitializeAsset....215
2.8.2.2.4 AssetReconstituted....216
2.8.2.2.5 AssetKilled....217
2.9 THE TERMINAL PROCESS....218
2.9.1 Command Line Interface....219
2.9.1.1 main.c....219
2.9.1.1.1 Terminal_Handler....219
2.9.1.1.2 Quit_Handler....219

2.9.1.1.3 main....219
2.9.1.2 cmd.c....220
2.9.1.2.1 cmd_send_breach....220
2.9.1.2.2 cmd_clock....221
2.9.1.2.3 cmd_disable....221
2.9.1.2.4 cmd_exit....222
2.9.1.2.5 cmd_list....222
2.9.1.2.6 cmd_perf....223
2.9.1.2.7 cmd_restart....223
2.9.1.2.8 cmd_restore....224
2.9.1.2.9 cmd_save....224
2.9.1.2.10 cmd_status_all....224
2.9.1.2.11 cmd_status_bumper....225
2.9.1.2.12 cmd_status_ipc....225
2.9.1.2.13 cmd_status_net....226
2.9.1.2.14 cmd_status_process....226
2.9.1.2.15 cmd_status_sim....227
2.9.1.2.16 cmd_status_sims....227
2.9.1.2.17 cmd_status_vid....228
2.9.1.2.18 cmd_status_yumm....228
2.9.1.2.19 cmd_trace....229
2.9.1.3 parser.c....229
2.9.1.3.1 parser_init....230
2.9.1.3.2 parser_restore_term....230
2.9.2 Clock Reset Command....230
2.9.2.1 clock.c....230
2.9.2.1.1 ClockCommand....231
2.9.2.1.2 LocalToGMTTime....231
2.9.3 InterProcess Communication Status Command....232
2.9.3.1 ipc.c....232
2.9.3.1.1 ReportIPCStatus....232
2.9.3.1.2 DisplayShmStatus....233
2.9.3.1.3 DisplaySemStatus....233
2.9.3.1.4 DisplayMsgQStatus....234
2.9.4 Performance Monitor Command....235
2.9.4.1 perf.c....235
2.9.4.1.1 ReportPerformance....235
2.9.5 Process Monitor Command....235
2.9.5.1 process.c....236
2.9.5.1.1 ProcessStatus....236
2.9.6 Status Report Command....236
2.9.6.1 status.c....237
2.9.6.1.1 SimStatusReport....237

2.9.6.1.2 VehicleStatusReport....238
2.9.7 Simulator Save/Restore....238
2.9.7.1 vehicle.c....238
2.9.7.1.1 SimListSaves....239
2.9.7.1.2 SimSaveAll....239
2.9.7.1.3 SimRestoreAll....240
2.9.7.1.4 VehFromSimNum....241
2.9.7.1.5 VehFromBumper....241
2.9.7.1.6 ValidSimNum....242
2.9.7.1.7 ValidBumperNumber....242
2.10 MASSCOMP COMMUNICATION SOFTWARE....243
2.10.1 ATSend Process....243
2.10.1.1 send.c....244
2.10.1.1.1 main....244
2.10.1.1.2 ProcessMessage....244
2.10.2 ATRecv Process....245
2.10.2.1 recv.c....245
2.10.1.2.1 main....245
2.11 THE BRIDGE CONSOLE....247
2.11.1 Bridge Program....247
2.11.1.1 HexDigit....247
2.11.1.2 BinNibble....248
2.11.1.3 QueueARead....248
2.11.1.4 QueueAWrite....249
2.11.1.5 QueueSWrite....249
2.11.1.6 EncodeFrame....250
2.11.1.7 ValidFrame....250
2.11.1.8 DecodeFrame....251
2.11.1.9 QueryFrame....252
2.11.1.10 SendAnswer....252
2.11.1.11 StartOfFrame....253
2.11.1.12 AppendByte....253
2.11.1.13 EndOfFrame....253
2.11.1.14 ChkSerIn....254
2.11.1.15 ChkSerOut....255
2.11.1.16 SetUp....255
2.11.1.17 Bridge....256
2.12 APPLETALK NETWORK SOFTWARE....258
2.13 THE SCC CONSOLE....260
2.13.1 Simulating Command Post Vehicles....261
2.13.1.1 adminlog.c....261
2.13.1.1.1 LoadCannedALOC....261
2.13.1.1.2 UploadALOCParameters....261
2.13.1.1.3 ALOCInitFetch....262
2.13.1.1.4 ALOCReconstFetch....262

2.13.1.1.5 ALOCEvent 263
2.13.1.2 toc.c 263
2.13.1.2.1 LoadCannedTOC 264
2.13.1.2.2 UploadTOCParameters 264
2.13.1.2.3 TOCInitFetch 265
2.13.1.2.4 TOCReconstFetch 265
2.13.1.2.5 TOCEvent 265
2.13.2 Allocation, Placement and Reconstitution of Vehicle Simulators 266
2.13.2.1 alloc.c 266
2.13.2.1.1 SetUpAlloc 267
2.13.2.1.2 AllocTableFetch 268
2.13.2.1.3 AllocTableSelect 268
2.13.2.1.4 AllocTableEvent 269
2.13.2.1.5 AllocEntryFetch 269
2.13.2.1.6 AllocEntryEvent 270
2.13.2.2 place.c 271
2.13.2.2.1 SetUpPlace 271
2.13.2.2.2 PlaceSelectFetch 272
2.13.2.2.3 PlaceSelectEvent 272
2.13.2.2.4 PlaceTableFetch 273
2.13.2.2.5 PlaceComplete 273
2.13.2.2.6 PlaceTableSelect 273
2.13.2.2.7 PlaceTableEvent 274
2.13.2.2.8 SimulatorPlaced 275
2.13.3 Simulation of Combat Service Support 275
2.13.3.1 css.c 275
2.13.3.1.1 LoadCannedCSS 277
2.13.3.1.2 CSSTypeFetch 277
2.13.3.1.3 CSSTypeBranch 278
2.13.3.1.4 DefaultTrainsEvent 278
2.13.3.1.5 OrgTrainsBranch 279
2.13.3.1.6 UploadCSSParameters 279
2.13.3.1.7 CSSConfirmFetch 280
2.13.3.1.8 CSSConfirmEvent 280
2.13.3.2 depots.c 281
2.13.3.2.1 DepotConfirmFetch 281
2.13.3.2.2 DepotConfirmEvent 282
2.13.3.3 truckentry.c 282
2.13.3.3.1 ShowTruckDialog 284
2.13.3.3.2 UpdateAmmoLoadTotals 284
2.13.3.3.3 TruckEntryFetch 285
2.13.3.3.4 TruckEntryEvent 285

2.13.3.3.5 AmmoTableSelect 286
2.13.3.3.6 AmmoTransferComplete 287
2.13.3.4 trucktable.c 287
2.13.3.4.1 SetUpTrucks 288
2.13.3.4.2 TruckDefaultAll 289
2.13.3.4.3 TruckDefault 289
2.13.3.4.4 TruckDefaultAlignment 290
2.13.3.4.5 AmmoTruckTableFetch 290
2.13.3.4.6 AmmoTruckTableEvent 290
2.13.3.4.7 FuelTruckTableFetch 291
2.13.3.4.8 MaintTruckTableFetch 292
2.13.3.4.9 TruckTableComplete 292
2.13.3.4.10 TruckTableSelect 292
2.13.3.4.11 TruckTableHilite 293
2.13.3.4.12 TruckDrawNumber 293
2.13.3.4.13 TruckDrawAssignment 294
2.13.3.4.14 TruckDrawLocation 294
2.13.3.4.15 TruckDrawFuelLoad 295
2.13.3.4.16 TruckDrawAmmoLoad 296
2.13.3.5 truck.h 296
2.13.3.6 truck.c 297
2.13.3.6.1 UploadTruckParameters 297
2.13.3.6.2 DownloadTruckParameters 298
2.13.4 Simulation of Close Air and Fire Support 298
2.13.4.1 cas.c 298
2.13.4.1.1 LoadCannedCAS 299
2.13.4.1.2 UploadCASParameters 299
2.13.4.1.3 InitCASEvent 300
2.13.4.1.4 MoreSortiesFetch 300
2.13.4.1.5 MoreSortiesEvent 301
2.13.4.2 fse.c 301
2.13.4.2.1 LoadCannedFSE 303
2.13.4.2.2 InitArtyFetch 303
2.13.4.2.3 InitArtyEvent 303
2.13.4.2.4 InitArtyBranch 304
2.13.4.2.5 UploadFSEParameters 304
2.13.4.2.6 FSEConfirmFetch 305
2.13.4.2.7 FSEConfirmEvent 305
2.13.4.2.8 UploadBtryParameters 306
2.13.4.2.9 DownloadBtryParameters 306
2.13.4.2.10 ShowBtryReconstDialog 307
2.13.4.2.11 BtryReconstEvent 307
2.13.4.2.12 GetGunAzimuth 308

2.13.4.3 cew.h 309
2.13.4.4 cew.c 309
2.13.4.4.1 LoadDefaultCEWParameters 311
2.13.4.4.2 CEWAllocFetch 311
2.13.4.4.3 CEWAllocEvent 311
2.13.4.4.4 UploadCEWParameters 312
2.13.4.4.5 CEWConfirmFetch 313
2.13.4.4.6 CEWConfirmEvent 313
2.13.4.4.7 CEWReconstFetch 314
2.13.4.4.8 CEWReconstEvent 314
2.13.4.4.9 UpdateCEWAssetLocation 315
2.13.4.4.10 ShowCEWReconstDialog 315
2.13.5 Reconstitution of Simulated Vehicles 316
2.13.5.1 reconscr.h 316
2.13.5.2 reconscr.c 316
2.13.5.2.1 NewReconstElement 316
2.13.5.2.2 NewReconstVehicle 317
2.13.5.2.3 DrawReconstElement 318
2.13.5.2.4 DrawReconstVehicle 318
2.13.5.3 reconst.c 319
2.13.5.3.1 SimReconstComplete 320
2.13.5.3.2 TruckReconstStart 320
2.13.5.3.3 TruckReconstComplete 320
2.13.5.3.4 ReconstFetch 321
2.13.5.3.5 ReconstElementSelect 322
2.13.5.3.6 ReconstVehicleSelect 323
2.13.5.3.7 ReconstEvent 323
2.13.6 Simulating Gunnery Targets 324
2.13.6.1 target.h 324
2.13.6.2 target.c 324
2.13.6.2.1 NewTarget 325
2.13.6.2.2 RemoveTarget 325
2.13.6.2.3 TargetHit 326
2.13.6.2.4 ResetTargets 326
2.13.6.2.5 UploadTarget 327
2.13.6.3 tgtentry.c 328
2.13.6.3.1 ShowTarget 328
2.13.6.3.2 TargetEntryFetch 329
2.13.6.3.3 TargetEntryEvent 329
2.13.6.4 tgttable.c 330
2.13.6.4.1 SetUpTargets 330
2.13.6.4.2 ShowTargetTable 331
2.13.6.4.3 TargetTableSelect 331

2.13.6.4.4 TargetTableHilite 332
2.13.6.4.5 TargetDrawName 332
2.13.6.4.6 TargetDrawType 333
2.13.6.4.7 TargetDrawAppearance 333
2.13.6.4.8 TargetDrawLocation 334
2.13.6.4.9 TargetTableEvent 334
2.13.7 Visual Appearance of SCC User Interface 335
2.13.7.1 SCC Pictures 335
2.13.7.2 resource.h 335
2.13.8 Auxiliary Software 335
2.13.8.1 SCC.h 335
2.13.8.2 SCCMac.h 335
2.13.8.3 SCC #includes.c 336
2.13.8.4 SCCoptions.h 337
2.13.8.5 version.h 337
2.13.8.6 atalk.c 337
2.13.8.6.1 ProcessRequest 337
2.13.8.6.2 ProcessSimProblemRequest 338
2.13.8.6.3 DownloadTerrain 338
2.13.8.6.4 DownloadOptions 339
2.13.8.7 data.c 340
2.13.8.8 dialog.c 340
2.13.8.8.1 CheckMandatoryEvent 341
2.13.8.8.2 ZoomUpFromTableEntry 341
2.13.8.8.3 ZoomDownToTableEntry 342
2.13.8.9 displace.h 342
2.13.8.10 displace.c 342
2.13.8.10.1 CompleteDisplacement 344
2.13.8.10.2 DisplaceFetch 344
2.13.8.10.3 DisplaceEvent 344
2.13.8.10.4 DisplaceBranch 345
2.13.8.10.5 DisplaceSelectElement 345
2.13.8.10.6 DisplaceSelectUnit 346
2.13.8.10.7 NewDispElement 347
2.13.8.10.8 NewDispUnit 347
2.13.8.10.9 DrawDispElement 348
2.13.8.10.10 DrawDispUnit 349
2.13.8.10.11 DrawDispStatus 349
2.13.8.10.12 DrawDispETA 350
2.13.8.10.13 SetUpDisplacement 351
2.13.8.10.14 DisplaceDispatchFetch 351
2.13.8.10.15 DisplaceDispatchEvent 351
2.13.8.10.16 DisplaceHaltFetch 352

2.13.8.10.17 DisplaceHaltEvent....353
2.13.8.10.18 ComputeETA....353
2.13.8.10.19 comp_pct....354
2.13.8.10.20 UpdateDisplacement....355
2.13.8.10.21 NetworkDispatched....355
2.13.8.10.22 NetworkArrived....356
2.13.8.11 exercise.c....357
2.13.8.11.1 LoadCannedExercise....358
2.13.8.11.2 MCCRoleFetch....358
2.13.8.11.3 MCCRoleEvent....359
2.13.8.11.4 SelectTerrainUpdate....359
2.13.8.11.5 SelectTerrainFetch....360
2.13.8.11.6 SelectTerrainEvent....360
2.13.8.11.7 LoadOptionsPermitted....361
2.13.8.11.8 SetUpOptionalElements....361
2.13.8.11.9 PermitOptionalElements....361
2.13.8.11.10 OptElement1Fetch....362
2.13.8.11.11 OptElement2Fetch....362
2.13.8.11.12 OptElement2Event....362
2.13.8.11.13 CoRoleFetch....363
2.13.8.11.14 UploadExerciseParameters....364
2.13.8.11.15 StartConfirmEvent....364
2.13.8.12 file.c....365
2.13.8.12.1 LoadGunneryTargets....365
2.13.8.12.2 SaveGunneryTargets....366
2.13.8.12.3 WriteGunneryTargets....366
2.13.8.12.4 ReadGunneryTargets....367
2.13.8.12.5 WipeTargetTable....368
2.13.8.12.6 LoadPresets....368
2.13.8.12.7 SavePresets....369
2.13.8.13 init.c....370
2.13.8.13.1 InitOverviewFetch....371
2.13.8.13.2 InitOverviewEvent....371
2.13.8.13.3 InitOverviewBranch....371
2.13.8.14 lock.c....372
2.13.8.14.1 ToggleConsoleLock....372
2.13.8.14.2 LockDialogEvent....373
2.13.8.15 main.c....373
2.13.8.15.1 main....373
2.13.8.15.2 LoadCannedData....374
2.13.8.15.3 MainEventLoop....374
2.13.8.15.4 ProcessCommandShiftOption Sequence....375

2.13.8.16 master.c 375
2.13.8.16.1 PasswordFetch 376
2.13.8.16.2 PasswordEvent 377
2.13.8.16.3 BMOverviewFetch 377
2.13.8.16.4 BMOverviewEvent 378
2.13.8.16.5 StopEvent 378
2.13.8.17 setup.c 379
2.13.8.17.1 SetUp 379
2.13.8.18 string.c 380
2.13.8.18.1 Pstrcpy 380
2.14 THE PLACE (PLACEMENT) CONSOLE 381
2.14.1 Place Console Definitions 381
2.14.1.1 Place.h 381
2.14.1.2 PlaceMac.h 382
2.14.1.3 version.h 382
2.14.1.4 data.c 382
2.14.1.5 resource.h 382
2.14.2 Place Console Software 383
2.14.2.1 atalk.c 383
2.14.2.1.1 ProcessRequest 383
2.14.2.1.2 DownloadParameters 383
2.14.2.2 dialog.c 384
2.14.2.2.1 ZoomUpFromTableEntry 384
2.14.2.2.2 ZoomDownToTableEntry 385
2.14.2.3 main.c 385
2.14.2.3.1 main 385
2.14.2.3.2 LoadCannedTerrainMap 386
2.14.2.3.3 MainEventLoop 386
2.14.2.4 place.c 386
2.14.2.4.1 SetUpPlace 387
2.14.2.4.2 PlaceSelectFetch 387
2.14.2.4.3 PlaceSelectEvent 388
2.14.2.4.4 PlaceTableFetch 388
2.14.2.4.5 PlaceComplete 389
2.14.2.4.6 PlaceTableSelect 389
2.14.2.4.7 PlaceTableEvent 390
2.14.2.4.8 SimulatorAlloced 390
2.14.2.4.9 SimulatorPlaced 391
2.14.2.5 setup.c 391
2.14.2.5.1 SetUp 391
2.14.3 Appearance of User Interface 392
2.14.3.1 Place Pictures 392

2.15 THE ADMIN CONSOLE 393
2.15.1 Truck Load and Unload at Supply Depots 394
2.15.1.1 load.c 394
2.15.1.1.1 ShowLoadDialog 394
2.15.1.1.2 AmmoTruckLoadComplete 395
2.15.1.1.3 FuelTruckLoadFetch 395
2.15.1.1.4 FuelTruckLoadEvent 395
2.15.1.1.5 UpdateLoadDialog 396
2.15.1.1.6 CloseLoadDialog 396
2.15.2 Interface between Admin Console and MCC Host 397
2.15.2.1 atalk.c 397
2.15.2.1.1 ProcessRequest 397
2.15.2.1.2 ProcessResponse 398
2.15.2.1.3 ReportDispatch 398
2.15.2.1.4 ReportHalt 399
2.15.3 Model Travel of Supply Trucks 399
2.15.3.1 dispatch.c 399
2.15.3.1.1 ShowDispatchDialog 400
2.15.3.1.2 TruckDispatchFetch 401
2.15.3.1.3 TruckDispatchEvent 401
2.15.3.1.4 ComputeETA 402
2.15.3.1.5 UpdateDispatchLoad 402
2.15.3.1.6 CloseDispatchDialog 403
2.15.4 Admin Console Definitions 403
2.15.4.1 Admin.h 403
2.15.4.2 AdminMac.h 403
2.15.5 Admin Console Software 404
2.15.5.1 main.c 404
2.15.5.1.1 main 404
2.15.5.1.2 DownloadInitialData 405
2.15.5.1.3 MainEventLoop 405
2.15.5.2 setup.c 406
2.15.5.2.1 SetUp 406
2.15.5.3 demo.c 407
2.15.5.3.1 LoadCannedData 407
2.15.5.4 table.c 408
2.15.5.4.1 SetUpTruckTables 408
2.15.5.4.2 ShowTruckTables 409
2.15.5.4.3 TruckTableFetch 409
2.15.5.4.4 TruckTableEvent 409
2.15.5.4.5 TruckTableSelect 410
2.15.5.4.6 TruckDrawNumber 411
2.15.5.4.7 TruckDrawAssignment 412

2.15.5.4.8 TruckDrawAmmoLoad 412
2.15.5.4.9 TruckDrawFuelLoad 413
2.15.5.4.10 TruckDrawStatus 413
2.15.5.4.11 TruckDrawLocation 414
2.15.5.4.12 TruckDrawETA 414
2.15.5.4.13 TruckTableHilite 415
2.15.5.4.14 UpdateTruckTableRow 416
2.15.5.4.15 UpdateButtons 416
2.15.5.4.16 SelectTruckTableRow 417
2.15.5.5 halt.c 418
2.15.5.5.1 ShowHaltDialog 418
2.15.5.5.2 TruckHaltFetch 419
2.15.5.5.3 TruckHaltEvent 419
2.15.5.5.4 UpdateHaltLocation 420
2.15.5.5.5 UpdateHaltLoad 420
2.15.5.5.6 CloseHaltDialog 421
2.15.5.6 data.c 421
2.15.5.7 models.c 422
2.15.5.7.1 InitTruckState 422
2.15.5.7.2 CheckTimers 423
2.15.5.7.3 UpdateTruckLocation 423
2.15.5.7.4 DisableTruck 424
2.15.5.7.5 EnableTruck 425
2.15.5.7.6 ScheduleDisableEvent 425
2.15.5.7.7 NotifyStatusChange 426
2.15.5.7.8 UpdatePopUpLoad 427
2.15.5.7.9 DiscardPopUpDialog 427
2.15.6 Appearance of Admin User Interface 427
2.15.6.1 Admin Pictures 427
2.16 THE MAINT (MAINTENANCE) CONSOLE 428
2.16.1 Interface between Maint Console and MCC Host 429
2.16.1.1 atalk.c 429
2.16.1.1.1 ProcessRequest 429
2.16.1.1.2 ProcessResponse 430
2.16.1.1.3 ReportDispatch 430
2.16.1.1.4 ReportHalt 431
2.16.1.1.5 ReportRepair 431
2.16.2 Model Travel of Maint Teams 432
2.16.2.1 recover.c 432
2.16.2.1.1 ShowRecoverDialog 432
2.16.2.1.2 RecoverFetch 433
2.16.2.1.3 RecoverEvent 433
2.16.2.1.4 ComputeETA 434

2.16.2.1.5 CloseRecoverDialog 434
2.16.3 Maint Console Definitions 435
2.16.3.1 Maint.h 435
2.16.3.2 MaintMac.h 435
2.16.4 Maint Console Software 436
2.16.4.1 main.c 436
2.16.4.1.1 main 436
2.16.4.1.2 DownloadInitialData 437
2.16.4.1.3 MainEventLoop 437
2.16.4.2 setup.c 438
2.16.4.2.1 SetUp 438
2.16.4.3 resource.h 439
2.16.4.4 demo.c 439
2.16.4.4.1 LoadCannedData 439
2.16.4.5 teamtable.c 440
2.16.4.5.1 SetUpTeamTable 440
2.16.4.5.2 ShowTeamTable 440
2.16.4.5.3 TeamTableFetch 441
2.16.4.5.4 TeamTableEvent 441
2.16.4.5.5 TeamTableSelect 442
2.16.4.5.6 TeamDrawNumber 442
2.16.4.5.7 TeamDrawAssignment 443
2.16.4.5.8 TeamDrawStatus 444
2.16.4.5.9 TeamDrawLocation 444
2.16.4.5.10 TeamDrawETA 445
2.16.4.5.11 TeamTableHilite 445
2.16.4.5.12 UpdateTeamTableRow 446
2.16.4.5.13 UpdateTeamTableButtons 447
2.16.4.6 dispatch.c 448
2.16.4.6.1 ShowDispatchDialog 448
2.16.4.6.2 TeamDispatchFetch 449
2.16.4.6.3 TeamDispatchEvent 449
2.16.4.6.4 ComputeETA 450
2.16.4.6.5 CloseDispatchDialog 450
2.16.4.7 start.c 451
2.16.4.7.1 ShowStartDialog 451
2.16.4.7.2 StartFetch 452
2.16.4.7.3 StartEvent 452
2.16.4.7.4 StartSelect 453
2.16.4.7.5 StartHilite 453
2.16.4.7.6 StartDrawDescription 454
2.16.4.7.7 StartDrawClass 454
2.16.4.7.8 StartDrawDuration 455

2.16.4.7.9 UpdateStartDialog....455
2.16.4.7.10 UpdateStartButton....456
2.16.4.7.11 CloseStartDialog....456
2.16.4.7.12 NewRepair....457
2.16.4.8 repairtable.c....457
2.16.4.8.1 SetUpRepairTable....458
2.16.4.8.2 ShowRepairTable....458
2.16.4.8.3 RepairTableFetch....459
2.16.4.8.4 RepairTableEvent....459
2.16.4.8.5 RepairTableSelect....459
2.16.4.8.6 RepairTableDrawVehicle....460
2.16.4.8.7 RepairTableDrawLocation....460
2.16.4.8.8 RepairTableDrawDisabled....461
2.16.4.8.9 RepairTableDrawDescription....461
2.16.4.8.10 RepairTableDrawETC....462
2.16.4.8.11 RepairTableDrawState....463
2.16.4.8.12 UpdateRepairTableRow....463
2.16.4.9 cancel.c....464
2.16.4.9.1 ShowCancelDialog....464
2.16.4.9.2 RepairCancelFetch....464
2.16.4.9.3 RepairCancelEvent....465
2.16.4.9.4 CloseCancelDialog....465
2.16.4.9.5 CancelRepair....466
2.16.4.10 halt.c....466
2.16.4.10.1 ShowHaltDialog....467
2.16.4.10.2 TeamHaltFetch....467
2.16.4.10.3 TeamHaltEvent....468
2.16.4.10.4 UpdateHaltLocation....468
2.16.4.10.5 CloseHaltDialog....469
2.16.4.11 model.c....469
2.16.4.11.1 InitTeamState....469
2.16.4.11.2 CheckTimers....470
2.16.4.11.3 UpdateTeamLocation....470
2.16.4.11.4 RepairCompleted....471
2.16.4.11.5 DisableTeam....472
2.16.4.11.6 EnableTeam....472
2.16.4.11.7 ScheduleDisableEvent....473
2.16.4.11.8 NotifyStateChange....473
2.16.4.11.9 DiscardPopUpDialog....474
2.16.4.12 data.c....475
2.16.5 Appearance of Maint User Interface....476
2.16.5.1 Maint Pictures....476

2.17 THE FSE (FIRE SUPPORT ELEMENT) CONSOLE 477
2.17.1 User Interface for Controlling Indirect Fire Missions 478
2.17.1.1 adjust.c 478
2.17.1.1.1 ShowAdjust 478
2.17.1.1.2 AdjustEvent 479
2.17.1.1.3 ComputeAdjustment 479
2.17.1.1.4 ApproximateSin 480
2.17.1.1.5 ApproximateCos 480
2.17.1.2 fire.c 481
2.17.1.2.1 SetUpFireMission 481
2.17.1.2.2 GetFireDialogInfo 482
2.17.1.2.3 OpenFireMission 482
2.17.1.2.4 ShowFireMission 483
2.17.1.2.5 FireEvent 483
2.17.1.2.6 LoadFireBuffers 484
2.17.1.2.7 UpdateFireDialog 484
2.17.1.2.8 UpdateFireState 485
2.17.1.2.9 UpdateOTDirection 485
2.17.1.2.10 EndFireMission 486
2.17.1.3 ffe.c 487
2.17.1.3.1 ShowFFE 487
2.17.1.3.2 ShowFFEWithAdjust 487
2.17.1.3.3 FFEEvent 488
2.17.1.3.4 CommenceFFE 489
2.17.1.3.5 CancelAtCommand 489
2.17.1.3.6 CheckFiring 489
2.17.1.4 fpf.c 489
2.17.1.4.1 SetUpFPFMission 490
2.17.1.4.2 GetFPFDialogInfo 491
2.17.1.4.3 ShowNewFPFMission 491
2.17.1.4.4 OpenFPFMission 491
2.17.1.4.5 ShowFPFMission 492
2.17.1.4.6 FPFEvent 492
2.17.1.4.7 ValidFPFTarget 493
2.17.1.4.8 LoadFPFBuffers 494
2.17.1.4.9 UpdateFPFDialog 494
2.17.1.5 new.c 495
2.17.1.5.1 ShowNewFireMission 496
2.17.1.5.2 NewMissionEvent 496
2.17.1.6 target.c 497
2.17.1.6.1 ValidTgtLocation 497
2.17.2 Status of Artillery Batteries 498
2.17.2.1 battery.c 498

2.17.2.1.1 InstallDrawRoutine....498
2.17.2.1.2 SetUpBattery....499
2.17.2.1.3 ShowBattery....499
2.17.2.1.4 BatteryEventHandler....500
2.17.2.1.5 DrawBatteryText....500
2.17.2.1.6 DrawBatteryRect....501
2.17.2.1.7 UpdateBatteryDisplay....502
2.17.2.1.8 UpdateAmmoDisplay....502
2.17.2.2 status.c....503
2.17.2.2.1 SetUpStatusWindow....504
2.17.2.2.2 ShowStatusWindow....505
2.17.2.2.3 DrawStatusWindow....505
2.17.2.2.4 StatusEventHandler....506
2.17.2.2.5 RedrawMissionIcon....506
2.17.2.2.6 RedrawBatteryIcon....507
2.17.2.2.7 ZoomMissionIcon....508
2.17.2.2.8 ZoomBatteryIcon....508
2.17.3 Displacement of Artillery Batteries....509
2.17.3.1 displace.c....509
2.17.3.1.1 ShowDisplace....509
2.17.3.1.2 ThrowDisplace....510
2.17.3.1.3 DisplaceFetch....511
2.17.3.1.4 UpdateReleasePoint....511
2.17.3.1.5 DisplaceEvent....512
2.17.3.1.6 UndoDisplace....512
2.17.3.1.7 AbortFetch....513
2.17.3.1.8 AbortEvent....513
2.17.3.1.9 RedrawUnitLocation....514
2.17.3.2 move.c....514
2.17.3.2.1 LocalUnitDispatched....514
2.17.3.2.2 LocalUnitArrived....515
2.17.3.2.3 RemoteUnitDispatched....516
2.17.3.2.4 RemoteUnitArrived....517
2.17.3.2.5 InterpolatePoints....517
2.17.3.2.6 UpdateUnitLocation....518
2.17.4 Dialog Lists of Scheduled Missions....519
2.17.4.1 firetarget.c....519
2.17.4.1.1 NewFireTarget....520
2.17.4.1.2 SetUpFireTable....520
2.17.4.1.3 LookupFireTarget....521
2.17.4.1.4 ShowFireTable....521
2.17.4.1.5 FireTableSelect....521
2.17.4.1.6 TableDrawNumber....522

2.17.4.1.7 FireTableDrawLocation....523
2.17.4.1.8 FireTableDrawDescription....523
2.17.4.1.9 FireTableDrawRemarks....524
2.17.4.1.10 InstallFireTarget....524
2.17.4.1.11 ShowFireTarget....525
2.17.4.1.12 FireEntryEvent....525
2.17.4.1.13 ValidNewFireTarget....526
2.17.4.1.14 UndoFireTarget....526
2.17.4.1.15 UnhookFireTarget....527
2.17.4.2 fpftarget.c....527
2.17.4.2.1 NewFPFTarget....528
2.17.4.2.2 SetUpFPFTable....529
2.17.4.2.3 ShowFPFTable....529
2.17.4.2.4 FPFTableSelect....530
2.17.4.2.5 FPFTableDrawNumber....530
2.17.4.2.6 FPFTableDrawLeft....531
2.17.4.2.7 FPFTableDrawRight....531
2.17.4.2.8 FPFTableDrawDescription....532
2.17.4.2.9 InstallFPFTarget....532
2.17.4.2.10 ShowFPFTarget....533
2.17.4.2.11 FPFEntryEvent....533
2.17.4.2.12 ValidNewFPFTarget....534
2.17.4.2.13 UndoFPFTarget....534
2.17.4.3 schedule.c....535
2.17.4.3.1 NewSchedMission....536
2.17.4.3.2 SetUpSchedTable....536
2.17.4.3.3 ShowSchedTable....537
2.17.4.3.4 SchedTableSelect....537
2.17.4.3.5 SchedTableDrawTime....538
2.17.4.3.6 SchedTableDrawTarget....538
2.17.4.3.7 SchedTableDrawDescription....539
2.17.4.3.8 SchedTableDrawUnits....539
2.17.4.3.9 SchedTableDrawRounds....540
2.17.4.3.10 SchedTableDrawStatus....540
2.17.4.3.11 ShowSchedMission....541
2.17.4.3.12 SchedEntryEvent....542
2.17.4.3.13 ValidTime....542
2.17.4.3.14 UndoSchedMission....543
2.17.4.3.15 StartSchedMission....543
2.17.4.3.16 SetSchedMission....544
2.17.5 Model Operation of Artillery Batteries....545
2.17.5.1 model.c....545
2.17.5.1.1 CheckTimers....545

2.17.5.1.2 FireVolley 546
2.17.5.1.3 SendFireRequest 547
2.17.5.1.4 SendFPFRequest 548
2.17.5.1.5 ApplyToBatteries 548
2.17.5.1.6 StartMissionBattery 549
2.17.5.1.7 EndMissionBattery 549
2.17.5.1.8 CheckBattery 550
2.17.5.1.9 BatterySelectionViable 550
2.17.5.1.10 BatteryViable 551
2.17.5.1.11 TimeOfFlight 552
2.17.5.2 units.c 552
2.17.5.2.1 EstablishBatterySelection 553
2.17.5.2.2 UpdateBatterySelection 553
2.17.5.2.3 WithdrawBattery 554
2.17.6 Appearance of FSE User Interface 555
2.17.6.1 FSE Pictures 555
2.17.6.2 resource.h 555
2.17.7 FSE Console Definitions 555
2.17.7.1 FSE.h 555
2.17.7.2 FSEMac.h 555
2.17.7.3 version.h 556
2.17.7.4 data.c 557
2.17.8 FSE Console Software 557
2.17.8.1 file.c 557
2.17.8.1.1 NewPresets 558
2.17.8.1.2 LoadPresets 558
2.17.8.1.3 SavePresets 559
2.17.8.1.4 GetConfig 560
2.17.8.1.5 SaveConfig 560
2.17.8.1.6 CheckConfig 561
2.17.8.1.7 SaveTerrainMap 562
2.17.8.1.8 ReadTerrainMap 562
2.17.8.1.9 SaveStatus 563
2.17.8.1.10 ReadStatus 564
2.17.8.1.11 SaveFireTargets 564
2.17.8.1.12 ReadFireTargets 565
2.17.8.1.13 WipeFireTable 566
2.17.8.1.14 SaveFPFTargets 566
2.17.8.1.15 ReadFPFTargets 567
2.17.8.1.16 WipeFPFTable 568
2.17.8.1.17 SaveSchedMissions 569
2.17.8.1.18 ReadSchedMissions 569
2.17.8.1.19 CheckSchedMission 570
2.17.8.1.20 WipeSchedTable 571

2.17.8.2 load.c 572
2.17.8.2.1 LoadCannedTerrainMap 572
2.17.8.2.2 LoadCannedStatus 572
2.17.8.2.3 LoadCannedFireTargets 573
2.17.8.2.4 LoadCannedFPFTargets 573
2.17.8.3 main.c 574
2.17.8.3.1 main 574
2.17.8.3.2 DownloadGunData 575
2.17.8.3.3 MainEventLoop 575
2.17.8.3.4 AdjustScreenState 576
2.17.8.3.5 ShowHelp 576
2.17.8.3.6 ProcessRequest 577
2.17.8.4 menu.c 577
2.17.8.4.1 SetUpMenus 578
2.17.8.4.2 AddMenuFile 578
2.17.8.4.3 FutureMissionTest 579
2.17.8.4.4 EnableMenus 579
2.17.8.4.5 MenuCommand 580
2.17.8.5 mission.c 581
2.17.8.5.1 NewMission 581
2.17.8.5.2 EndMission 582
2.17.8.5.3 OpenMission 582
2.17.8.5.4 ToggleBattery 583
2.17.8.5.5 UpdateMissionDisplay 583
2.17.8.5.6 UpdateMissionState 584
2.17.8.5.7 UpdateObserver 584
2.17.8.6 setup.c 585
2.17.8.6.1 SetUp 585
2.18 THE CAS (CLOSE AIR SUPPORT) CONSOLE 586
2.18.1 CAS Console Definitions 587
2.18.1.1 CAS.h 587
2.18.1.2 CASMac.h 587
2.18.1.3 version.h 587
2.18.1.4 data.c 588
2.18.1.5 resource.h 588
2.18.1.6 ditl.h 588
2.18.2 CAS Console Software 588
2.18.2.1 condition.c 588
2.18.2.1.1 SetUpCondition 589
2.18.2.1.2 CheckForCondition 589
2.18.2.1.3 RaiseCondition 590
2.18.2.1.4 ConditionFetch 590
2.18.2.1.5 ConditionEvent 591
2.18.2.2 dialog.c 591

2.18.2.2.1 SetUpDialog 593
2.18.2.2.2 ShowMission 593
2.18.2.2.3 ClearDialogsForMission 593
2.18.2.2.4 StuffTimeOnTarget 594
2.18.2.2.5 ZoomMissionDown 595
2.18.2.2.6 OnCallFetch 595
2.18.2.2.7 FutureEvent 596
2.18.2.2.8 ValidTOT 596
2.18.2.2.9 StuffHeldMission 597
2.18.2.2.10 HeldFetch 597
2.18.2.2.11 HeldEvent 598
2.18.2.2.12 PastFetch 598
2.18.2.2.13 PastEvent 599
2.18.2.3 limits.c 599
2.18.2.3.1 SetSortieLimits 599
2.18.2.3.2 AdditionalSorties 600
2.18.2.3.3 ShowSummary 600
2.18.2.3.4 CheckSortieLimits 601
2.18.2.3.5 SortiesScheduled 602
2.18.2.4 load.c 602
2.18.2.4.1 LoadCannedTerrainMap 603
2.18.2.4.2 LoadCannedLimits 603
2.18.2.4.3 LoadCannedSchedule 603
2.18.2.5 main.c 604
2.18.2.5.1 main 604
2.18.2.5.2 MainEventLoop 604
2.18.2.5.3 ProcessRequest 605
2.18.2.6 mission.c 606
2.18.2.6.1 NewMission 606
2.18.2.6.2 UpdateMission 606
2.18.2.6.3 HiliteMission 607
2.18.2.6.4 CancelMission 607
2.18.2.6.5 ClearHot 608
2.18.2.7 schedule.c 608
2.18.2.7.1 SetUpSchedule 609
2.18.2.7.2 ShowSchedule 610
2.18.2.7.3 ScheduleSelect 610
2.18.2.7.4 ScheduleHilite 610
2.18.2.7.5 ScheduleDrawType 611
2.18.2.7.6 ScheduleDrawTOT 611
2.18.2.7.7 ScheduleDrawLocation 612
2.18.2.7.8 ScheduleDrawDescription 612
2.18.2.7.9 ScheduleDrawSorties 613

2.18.2.7.10 ScheduleDrawResults....613
2.18.2.7.11 ScheduleDrawStatus....614
2.18.2.7.12 ScheduleEvent....614
2.18.2.8 setup.c....615
2.18.2.8.1 SetUp....615
2.18.3 Appearance of CAS Console User Interface....616
2.18.3.1 CAS Pictures....616
2.19 THE CEW (COMBAT ENGINEERING WORKSTATION) CONSOLE....617
2.19.1 CEW Console Definitions....618
2.19.1.1 version.h....618
2.19.1.2 CEWMac.h....618
2.19.1.3 CEW.h....618
2.19.1.4 SimnetTypes.h....619
2.19.1.5 resource.h....619
2.19.1.6 console.h....619
2.19.2 CEW Console Software....619
2.19.2.1 main.c....619
2.19.2.1.1 main....620
2.19.2.1.2 MainEventLoop....621
2.19.2.1.3 MacInits....622
2.19.2.1.4 MacQuit....623
2.19.2.1.5 ProcessRequest....624
2.19.2.1.6 GetTerrainBounds....625
2.19.2.2 menus.c....625
2.19.2.2.1 SetUpMenus....626
2.19.2.2.2 DoMenu....626
2.19.3 Dialog Lists of Assets....627
2.19.3.1 CannedStuff.c....627
2.19.3.1.1 LoadCannedTerrainMap....627
2.19.3.1.2 LoadCannedResources....627
2.19.3.2 CheckDialogs.c....628
2.19.3.2.1 isLegalGEMSSDensity....628
2.19.3.2.2 isLegalMinefield....628
2.19.3.2.3 ReviewDialogHandler....629
2.19.3.2.4 setEmplacementTime....630
2.19.3.2.5 CheckEmplacementDialog....632
2.19.3.2.6 setBreachTime....633
2.19.3.2.7 CheckBreachDialog....635
2.19.3.2.8 CheckMoveDialog....636
2.19.3.2.9 CheckMissionTypeDialog....638
2.19.3.2.10 CheckStatEmplacementDialog....639
2.19.3.2.11 CheckStatBreachDialog....640
2.19.3.2.12 CheckStatMoveDialog....640

2.19.3.2.13 CheckReviewDialog 641
2.19.3.2.14 DetermineMines 641
2.19.3.2.15 ReviewSelection.....642
2.19.3.3 DialogFieldSupport.c 644
2.19.3.3.1 CheckUTMString 644
2.19.3.3.2 CheckDTGString.....644
2.19.3.3.3 CheckvsCurrentTime 645
2.19.3.3.4 CheckvsCurrentTimeStr.....646
2.19.3.3.5 UTMToLongPt 647
2.19.3.3.6 GetDisTime 648
2.19.3.4 DoAbout.c.....648
2.19.3.4.1 SetParamText.....648
2.19.3.4.2 DoAbout.....649
2.19.3.4.3 NotImpDialog.....649
2.19.3.4.4 Notify.....650
2.19.3.4.5 MySound.....651
2.19.3.5 HotSpots.c.....651
2.19.3.5.1 indone.....652
2.19.3.5.2 MissionButton.....653
2.19.3.5.3 MissionSelection 653
2.19.3.5.4 StatusButton 654
2.19.3.5.5 MainHelpButton.....655
2.19.3.6 Refresh.c 656
2.19.3.6.1 PrintWindow.....656
2.19.3.7 ResItem.c.....657
2.19.3.7.1 ResourceItems 657
2.19.3.7.2 CheckList_Handler.....658
2.19.3.7.3 DeleteResourceLists.....659
2.19.3.8 ResStatusWindow.c 659
2.19.3.8.1 DoResourceStatus.....659
2.19.3.8.2 SetupResWindow 660
2.19.3.8.3 RefreshWindow 661
2.19.3.8.4 HeadingSetUp 661
2.19.3.8.5 DoneStatus 662
2.19.3.8.6 HelpStatus.....662
2.19.3.8.7 OtherEventLoop 663
2.19.3.8.8 ResCleanUp.....664
2.19.3.9 StatusSupport.c.....664
2.19.3.9.1 AssembleMissionString 665
2.19.3.9.2 AssembleResourceString.....666
2.19.3.9.3 ReplaceMissionField.....666
2.19.3.9.4 MarkField 667
2.19.3.9.5 ClearField 667

2.19.3.9.6 stccpy 668
2.19.3.10 Utils.c 668
2.19.3.10.1 LocalRectToGlobal 668
2.19.3.10.2 stxcpy 669
2.19.3.10.3 stpcpy 669
2.19.3.10.4 HiliteRect 670
2.19.3.10.5 OutLineRect 670
2.19.3.10.6 GetCurrentMonth 671
2.19.3.10.7 CEStringToDTG 671
2.19.3.10.8 CEDTGToString 672
2.19.4 Minefield Geometry 672
2.19.4.1 poly_area.c 672
2.19.4.1.1 poly_area 672
2.19.4.1.2 poly_area_loc 673
2.19.4.1.3 inside_poly 675
2.19.4.2 poly_area.h 675
2.19.4.3 seg_intrsct.c 676
2.19.4.3.1 seg_intrsct 676
2.19.4.3.2 check_x_ints 677
2.19.4.3.3 check_y_ints 678
2.19.4.3.4 reverse_param 680
2.19.4.3.5 print_int_code 680
2.19.4.4 seg_intrsct.h 681
2.19.5 CEW Objects 681
2.19.5.1 CheckList.c 681
2.19.5.1.1 CheckList::init 681
2.19.5.1.2 CheckList::AddRow 682
2.19.5.1.3 CheckList::SetRow 683
2.19.5.1.4 CheckList::GetRow 684
2.19.5.1.5 CheckList::SetState 684
2.19.5.1.6 CheckList::GetState 685
2.19.5.1.7 CheckList::ToggleState 686
2.19.5.2 CheckList.h 686
2.19.5.3 ComWindows.c 686
2.19.5.3.1 ComWindow::show 687
2.19.5.3.2 ComWindow::AddHotSpot 687
2.19.5.3.3 ComWindow::initial 687
2.19.5.3.4 ComWindow::CheckHotSpot 688
2.19.5.4 ComWindows.h 689
2.19.5.5 Dialog_Windows.c 689
2.19.5.5.1 DialogWindow::init 689
2.19.5.5.2 DialogWindow::handler 690
2.19.5.5.3 DialogWindow::activate 691

2.19.5.5.4 DialogWindow::quit....691
2.19.5.5.6 DialogWindow::close....692
2.19.5.5.7 DialogWindow::GetText....692
2.19.5.5.8 DialogWindow::SetText....693
2.19.5.5.9 DialogWindow::SelectText....693
2.19.5.5.10 DialogWindow::GetValue....694
2.19.5.5.11 DialogWindow::SetValue....694
2.19.5.5.12 DialogWindow::HiliteItem....695
2.19.5.5.13 DialogWindow::SetUserItem....696
2.19.5.5.14 DialogWindow::GetItemRect....696
2.19.5.5.15 DialogWindow::equal....697
2.19.5.5.16 DialogWindow::GetDialogPtr....698
2.19.5.5.17 DialogWindow::BringToFront....698
2.19.5.5.18 DialogOutlineItem....698
2.19.5.6 Dialog_Windows.h....699
2.19.5.7 MissionObj.c....699
2.19.5.7.1 MissionObj::init....700
2.19.5.7.2 MissionObj::edit....701
2.19.5.7.3 MissionObj::NewMission....702
2.19.5.7.4 MissionObj::show....703
2.19.5.7.5 MissionObj::UpdateMissionString....705
2.19.5.7.6 MissionObj::GetStartMoveTime....705
2.19.5.7.7 MissionObj::GetStartActivityTime....706
2.19.5.7.8 MissionObj::GetEndTime....706
2.19.5.7.9 MissionObj::SetStartMoveTime....706
2.19.5.7.10 MissionObj::SetStartActivityTime....707
2.19.5.7.11 MissionObj::SetEndTime....707
2.19.5.7.12 MissionObj::GetNumber....708
2.19.5.7.13 MissionObj::SetResStatus....708
2.19.5.7.14 MissionObj::UpdateResAssigned....709
2.19.5.7.15 MissionObj::CheckResStatus....709
2.19.5.7.16 MissionObj::UpdateResPosition....710
2.19.5.7.17 MissionObj::ResetMissionTime....710
2.19.5.7.18 MissionObj::DispatchResources....712
2.19.5.7.19 MissionObj::ArrivalResources....712
2.19.5.7.20 MissionObj::SetStatus....713
2.19.5.7.21 MissionObj::GetStatus....713
2.19.5.7.22 MissionObj::GetType....714
2.19.5.7.23 MissionObj::GetOrderType....714
2.19.5.7.24 MissionObj::SendStartMovement....714
2.19.5.7.25 MissionObj::SendStartActivity....715
2.19.5.7.26 MissionObj::SendEndMission....715
2.19.5.7.27 MissionObj::SendEmplacement....715

2.19.5.7.28 MissionObj::SendBreach 716
2.19.5.7.29 MissionObj::SendDispatch 717
2.19.5.7.30 MissionObj::SendArrival 717
2.19.5.8 MissionObj.h 718
2.19.5.9 ResourceObj.c 718
2.19.5.9.1 ResourceObj::init 718
2.19.5.9.2 ResourceObj::GetKind 719
2.19.5.9.3 ResourceObj::GetCCVNumber 720
2.19.5.9.4 ResourceObj::SetPosition 720
2.19.5.9.5 ResourceObj::GetMission 721
2.19.5.9.6 ResourceObj::SetMission 721
2.19.5.9.7 ResourceObj::GetPosition 721
2.19.5.9.8 ResourceObj::GetKph 722
2.19.5.9.9 ResourceObj::DistanceToSecs 722
2.19.5.9.10 ResourceObj::GetName 723
2.19.5.9.11 ResourceObj::SetStatus 723
2.19.5.9.12 ResourceObj::GetStatus 724
2.19.5.9.13 ResourceObj::show 724
2.19.5.10 ResourceObj.h 724
2.19.5.11 ServiceObj.c 725
2.19.5.11.1 ServiceObj::init 725
2.19.5.11.2 ServiceObj::AddService 725
2.19.5.11.3 ServiceObj::DeleteService 726
2.19.5.11.4 ServiceObj::CheckQs 726
2.19.5.11.5 ServiceObj::ShowQ 728
2.19.5.11.6 ServiceObj::UpdateQs 728
2.19.5.12 ServiceObj.h 728
2.19.5.13 Simple_Windows.c 728
2.19.5.13.1 Swindow::init 729
2.19.5.13.2 Swindow::show 729
2.19.5.13.3 Swindow::quit 730
2.19.5.13.4 Swindow::update 730
2.19.5.13.5 Swindow::hide 731
2.19.5.13.6 Swindow::IsVisible 731
2.19.5.13.7 Swindow::close 731
2.19.5.13.8 Swindow::equal 731
2.19.5.13.9 Swindow::BringToFront 732
2.19.5.13.10 Swindow::GetWindowPtr 732
2.19.5.14 Simple_Windows.h 733
2.19.5.15 TextList.c 733
2.19.5.15.1 TextList::init 733
2.19.5.15.2 TextList::ListUpdate 734
2.19.5.15.3 TextList::ListDispose 734

2.19.5.15.4 TextList::GetCellRect 734
2.19.5.15.5 TextList::GetCount 735
2.19.5.15.6 TextList::AddRow 735
2.19.5.15.7 TextList::DeleteRow 736
2.19.5.15.8 TextList::MarkRow 736
2.19.5.15.9 TextList::SetRow 737
2.19.5.15.10 TextList::GetRow 738
2.19.5.15.11 TextList::ListProc 739
2.19.5.15.12 TextList::List1Click 739
2.19.5.15.13 stxcpy 740
2.19.5.16 TextList.h 741
2.20 THE NETWORK COMMUNICATION LIBRARIES 742
2.20.1 libassoc 742
2.20.1.1 subscribe.c 744
2.20.1.1.1 AssocSubscribe 744
2.20.1.1.2 AssocUnsubscribe 745
2.20.1.1.3 AssocCurrentlySubscribed 745
2.20.1.1.4 AssocSubscribeWithMask 746
2.20.1.1.5 AssocUnsubscribeWithMask 747
2.20.1.1.6 AssocCurrentlySubscribedWith Mask 748
2.20.1.1.7 AssocCreateMCA 749
2.20.1.1.8 AssocCreateMCAWithMask 749
2.20.1.1.9 AddSubscription 750
2.20.1.1.10 DeleteSubscription 751
2.20.1.2 send.c 751
2.20.1.2.1 AssocSendDatagram 752
2.20.1.2.2 AssocPadBuffer 753
2.20.1.3 aggregate.c 753
2.20.1.3.1 AssocSendAggregate 753
2.20.1.4 transact.c 755
2.20.1.4.1 AssocSendTransact 755
2.20.1.4.2 AssocSendResponse 757
2.20.1.5 open.c 758
2.20.1.5.1 AssocOpen 758
2.20.1.5.2 AssocAttach 760
2.20.1.5.3 SetChannelDefaults 761
2.20.1.6 receive.c 761
2.20.1.6.1 AssocReceivePDU 761
2.20.1.7 block.c 763
2.20.1.7.1 AssocWaitForPDU 764
2.20.1.8 tick.c 765
2.20.1.8.1 AssocTickAssocLayer 765

2.20.1.8.2 UpdateTransactions 766
2.20.1.9 address.c 767
2.20.1.9.1 AssocGetSimAddress....767
2.20.1.10 error.c....768
2.20.1.10.1 AssocError....768
2.20.1.11 close.c 768
2.20.1.11.1 AssocClose....768
2.20.1.12 family.c....769
2.20.1.12.1 AssocSetProtocolFamily 769
2.20.1.13 mask.c....770
2.20.1.13.1 AssocSetSendMask....770
2.20.1.13.2 AssocGetRspMask....771
2.20.1.13.3 AssocSetRspMask 771
2.20.1.14 raw.c....772
2.20.1.14.1 AssocReceiveAssocPDU....772
2.20.1.15 who.c....774
2.20.1.15.1 AssocGetLastAddress 774
2.20.1.16 time_list.c 775
2.20.1.16.1 AssocAddToStartOfTimeList....775
2.20.1.16.2 AssocAddToEndOfTimeList....775
2.20.1.16.3 AssocDeleteFromTimeList....776
2.20.1.16.4 AssocMoveToEndOfTimeList....776
2.20.1.17 strtok.c....777
2.20.1.17.1 strtok....777
2.20.1.18 respondent.c 777
2.20.1.18.1 AssocInitResponse....778
2.20.1.18.2 AssocCacheResponse....778
2.20.1.18.3 AssocDeleteCachedResponse....779
2.20.1.18.4 AssocTimeOutOldResponses....779
2.20.1.18.5 AssocFindResponse 780
2.20.1.19 proc_rsp.c....781
2.20.1.19.1 AssocProcessResponsePDU....781
2.20.1.20 proc_req.c....782
2.20.1.20.1 AssocProcessRequestPDU....782
2.20.1.21 proc_dgram.c....783
2.20.1.21.1 AssocProcessDatagramPDU....783
2.20.1.22 params.c....784
2.20.1.22.1 upshift 784
2.20.1.22.2 AssocReadParams....784
2.20.1.22.3 ProcessSite....785
2.20.1.22.4 ProcessHost....786
2.20.1.22.5 ProcessMaxSubscriptions....786
2.20.1.22.6 ProcessInitDescriptors....787

2.20.1.22.7 ProcessAddlDescriptors 787
2.20.1.22.8 ProcessProtocolFamily 788
2.20.1.23 origin.c 788
2.20.1.23.1 AssocInitTransactions 788
2.20.1.23.2 AssocAddTransaction 789
2.20.1.23.3 AssocDeleteTransaction 790
2.20.1.23.4 AssocFindTransaction 790
2.20.1.23.5 AssocRescheduleTransaction 791
2.20.1.24 free_list.c 791
2.20.1.24.1 AssocCreateFreeList 791
2.20.1.24.2 AssocGrowFreeList 792
2.20.1.24.3 AssocGetDescriptor 793
2.20.1.24.4 AssocFreeDescriptor 794
2.20.1.25 bucket.c 794
2.20.1.25.1 AssocAddToBucket 794
2.20.1.25.2 AssocDeleteFromBucket 795
2.20.1.25.3 AssocBucketLookup 795
2.20.1.26 assoc.h 796
2.20.1.27 assoc_lcl.h 796
2.20.1.28 defaults.h 797
2.20.1.29 AssocDef 797
2.20.2 libnetif 798
2.20.2.1 net_ctl.c 798
2.20.2.1.1 net_alive 798
2.20.2.1.2 net_res 799
2.20.2.1.3 net_settimeout 799
2.20.2.1.4 net_iocont 800
2.20.2.1.5 net_nopened 801
2.20.2.1.6 net_loopback 802
2.20.2.2 net_addr.c 802
2.20.2.2.1 net_addr_compare 802
2.20.2.2.2 net_addr_bin_to_str 803
2.20.2.2.3 net_addr_str_to_bin 803
2.20.2.2.4 net_getaddr 804
2.20.2.2.5 net_zeroaddr 805
2.20.2.2.6 net_addr_format_convert 805
2.20.2.3 net_clos.c 805
2.20.2.3.1 net_close 805
2.20.2.4 net_flus.c 807
2.20.2.4.1 net_flush 807
2.20.2.5 net_stat.c 808
2.20.2.5.1 net_get_statistics 808
2.20.2.5.2 net_zero_statistics 809

2.20.2.5.3 net_stat_string 810
2.20.2.5.4 net_print_statistics....810
2.20.2.6 net_info.c....811
2.20.2.6.1 net_hostbuf_info 811
2.20.2.6.2 net_sharebuf_info 812
2.20.2.6.3 net_bufs 813
2.20.2.6.4 net_interface_type 813
2.20.2.6.5 net_syserror_info....814
2.20.2.6.6 net_version....814
2.20.2.7 net_mode.c....815
2.20.2.7.1 net_norm 815
2.20.2.7.2 net_intloop 816
2.20.2.7.3 net_extloop....817
2.20.2.7.4 net_prom 817
2.20.2.7.5 do_mode_cmd 818
2.20.2.7.6 do_mode_cmd_cmc....819
2.20.2.7.7 do_mode_cmd_147 819
2.20.2.8 net_time.c 820
2.20.2.8.1 net_gettime 820
2.20.2.8.2 net_settime 821
2.20.2.8.3 net_current_time....821
2.20.2.8.4 net_init_time 822
2.20.2.8.5 net_stomp_time....823
2.20.2.8.6 net_heartbeat....823
2.20.2.8.7 net_device_base 824
2.20.2.9 net_load.c 825
2.20.2.9.1 net_load....825
2.20.2.9.2 net_unload....828
2.20.2.10 net_open.c....832
2.20.2.10.1 net_open....832
2.20.2.10.2 net_get_parameters....834
2.20.2.10.3 net_set_parameterss....834
2.20.2.10.4 open_cmc....835
2.20.2.10.5 open_147....836
2.20.2.10.6 get_type....836
2.20.2.10.7 get_device_number....837
2.20.2.11 net_mca.c....838
2.20.2.11.1 net_add_mca 838
2.20.2.11.2 net_del_mca....838
2.20.2.11.3 net_init_mca....839
2.20.2.12 net_stam.c....839
2.20.2.12.1 net_stamp_enable....840
2.20.2.12.2 net_stamp_disable....840

2.20.2.12.3 net_get_timestamp.....841
2.20.2.12.4 net_put_timestamp.....841
2.20.2.13 net_run.c.....842
2.20.2.13.1 net_run.....842
2.20.2.13.2 net_stop.....843
2.20.2.14 net_orecv.c.....843
2.20.2.14.1 net_recv.....843
2.20.2.14.2 recv_cmc.....845
2.20.2.14.3 recv_147.....846
2.20.2.14.4 net_get_next_packet.....846
2.20.2.14.5 net_release_next_packet.....847
2.20.2.14.6 net_get_recv.....847
2.20.2.14.7 net_release_recv.....848
2.20.2.15 net_osend.c.....849
2.20.2.15.1 net_send.....849
2.20.2.15.2 send_cmc.....850
2.20.2.15.3 send_147.....851
2.20.2.15.4 net_get_send.....852
2.20.2.15.5 net_release_send.....852
2.20.2.16 net_reg.c.....853
2.20.2.16.1 net_reg_read.....853
2.20.2.16.2 net_reg_write.....854
2.20.2.17 net_recv.c.....855
2.20.2.17.1 net_reset_lock.....855
2.20.2.17.2 net_rcv.....855
2.20.2.17.3 net_get_rcv_to_addr.....856
2.20.2.17.4 net_get_rcv_from_addr.....857
2.20.2.17.5 net_get_rcv_type.....857
2.20.2.17.6 recv_cmc_8023.....858
2.20.2.17.7 recv_147_8023.....859
2.20.2.17.8 wait_for_full_ring_element.....860
2.20.2.17.9 net_get_rcv.....860
2.20.2.17.10 net_release_rcv.....861
2.20.2.18 net_type.c.....862
2.20.2.18.1 net_add_type.....862
2.20.2.18.2 net_init_type.....863
2.20.2.19 net_send.c.....863
2.20.2.19.1 wait_for_empty_ring_element.....863
2.20.2.19.2 net_snd.....864
2.20.2.19.3 send_cmc_8023.....865
2.20.2.19.4 send_147_8023.....866
2.20.2.19.5 net_set_snd_from_addr.....867
2.20.2.19.6 net_set_snd_type.....867

2.20.2.19.7 net_get_snd 868
2.20.2.19.8 net_release_snd....869
2.20.2.20 net_acce.c 870
2.20.2.20.1 net_access 870
2.20.2.20.2 access_cmc....873
2.20.2.20.3 access_147 873
2.20.2.21 net_stuf.c....874
2.20.2.21.1 get_locks 874
2.20.2.21.2 map_buffers....874
2.20.2.21.3 map_enp....876
2.20.2.21.4 unmap_enp....877
2.20.2.21.5 unmap_buffers....878
2.20.2.21.6 unget_locks 879
2.20.2.22 net_data.c....879
2.20.2.23 pktq.h....879
2.20.2.24 network.h 879
2.20.2.25 netlock.h....879
2.20.2.26 libnet.h....880
2.20.2.27 network.def 880
2.21 THE MCC LIBRARIES....882
2.21.1 libmcc....882
2.21.1.1 atatp.c....882
2.21.1.1.1 ATPSendRequest....883
2.21.1.1.2 ATPSendResponse....883
2.21.1.1.3 ATPTimeout 884
2.21.1.1.4 ATPRequest....884
2.21.1.1.5 ATPGetRequest 885
2.21.1.1.6 ATPResponse....885
2.21.1.1.7 ATReceive....886
2.21.1.2 atbuf.c 887
2.21.1.2.1 ATAllocBuffer....887
2.21.1.2.2 ATFreeBuffer....888
2.21.1.3 atnbp.c....888
2.21.1.3.1 NBPLookup 889
2.21.1.3.2 NBPReceive 889
2.21.1.4 atskt.c....890
2.21.1.4.1 ATAllocSocket....890
2.21.1.4.2 ATFreeSocket....891
2.21.1.4.3 SendPacket....892
2.21.1.5 azimuth.c....892
2.21.1.5.1 mil_to_fixedpt 892
2.21.1.5.2 fixedpt_to_mil 893
2.21.1.6 ccv.c 893

2.21.1.6.1 DisplayCCV....893
2.21.1.6.2 HideCCV....894
2.21.1.7 console.c....894
2.21.1.7.1 OpenHostSocket....895
2.21.1.7.2 ActivateConsole....895
2.21.1.7.3 ATPTransact....896
2.21.1.7.4 ATPPut....897
2.21.1.7.5 ATPPutComplete....897
2.21.1.7.6 SendConsoleResponse....898
2.21.1.7.7 ConsoleReleaseArrived....898
2.21.1.8 cew.c....898
2.21.1.8.1 GetCEWParameters....898
2.21.1.9 data.c....899
2.21.1.10 decode.c....900
2.21.1.10.1 DecodeHullToWorldMatrix....900
2.21.1.10.2 DecodeRotationVector....900
2.21.1.11 error.c....901
2.21.1.11.1 InitErrorHandling....901
2.21.1.11.2 SendErrorReport....902
2.21.1.11.3 MCC_SystemError....902
2.21.1.11.4 VehicleIDToTrailer....903
2.21.1.12 faad.c....903
2.21.1.12.1 EncodeVehicleStatusFAAD....904
2.21.1.12.2 DecodeVehicleStatusFAAD....904
2.21.1.12.3 TotalFuelFAAD....904
2.21.1.13 fqueue.c....904
2.21.1.13.1 Hash_Get_Entry....905
2.21.1.13.2 Alloc_Free....905
2.21.1.13.3 FQueue_Create....906
2.21.1.13.4 FQueue_Insert....907
2.21.1.13.5 FQueue_Remove....907
2.21.1.13.6 FQueue_Scan....908
2.21.1.13.7 FQueue_Retrieve....908
2.21.1.13.8 FQueue_Dump_Bucket....909
2.21.1.13.9 FQueue_Analyze....910
2.21.1.14 fred.c....910
2.21.1.14.1 EncodeVehicleStatusFRED....910
2.21.1.14.2 DecodeVehicleStatusFRED....911
2.21.1.14.3 fred_encode_failures....911
2.21.1.15 generic.c....911
2.21.1.15.1 TotalFuelGeneric....911
2.21.1.16 grid.c....912
2.21.1.16.1 ProcessGridInfoRequest....912

2.21.1.17 guise.c 913
2.21.1.17.1 GetGuises 913
2.21.1.18 hash.c 914
2.21.1.18.1 vList_Hash 914
2.21.1.18.2 vStatusList_Hash 914
2.21.1.19 init.c 915
2.21.1.19.1 InitializeProcess 915
2.21.1.19.2 TraceHandler 916
2.21.1.19.3 TraceMessage 916
2.21.1.19.4 InitUnit 917
2.21.1.20 m1.c 917
2.21.1.20.1 EncodeVehicleStatusM1 917
2.21.1.20.2 DecodeVehicleStatusM1 918
2.21.1.20.3 TotalFuelM1 918
2.21.1.20.4 m1_encode_failures 918
2.21.1.21 m2.c 919
2.21.1.21.1 EncodeVehicleStatusM2 919
2.21.1.21.2 DecodeVehicleStatusM2 919
2.21.1.21.3 TotalFuelM2 920
2.21.1.21.4 m2_encode_failures 920
2.21.1.22 net.c 921
2.21.1.22.1 SAF_NET_SND_KLUDGE 921
2.21.1.22.2 set_g_ethernet_two_packets 921
2.21.1.22.3 OpenNetworkInterface 922
2.21.1.22.4 CloseNetworkInterface 922
2.21.1.23 radio.c 922
2.21.1.23.1 InitRadios 922
2.21.1.23.2 RemoveRadios 923
2.21.1.23.3 TurnRadioOn 923
2.21.1.23.4 TurnRadioOff 923
2.21.1.24 segintr.c 924
2.21.1.24.1 seg_intrsct 924
2.21.1.24.2 check_x_ints 924
2.21.1.24.3 check_y_ints 926
2.21.1.24.4 reverse_param 927
2.21.1.25 sim.c 927
2.21.1.25.1 ProcessSimExistsRequest 928
2.21.1.25.2 ProcessSimInitRequest 928
2.21.1.25.3 SimGetSimType 929
2.21.1.25.4 ProcessSimQueryRequest 930
2.21.1.25.5 GetCompanyFromOrg 931
2.21.1.25.6 GetOldStyleCompany 931
2.21.1.25.7 PutCompanyInSim 932

2.21.1.26 spawn.c....932
2.21.1.26.1 SpawnProcess....932
2.21.1.26.2 LookupProcessIdentifier....933
2.21.1.26.3 LookupProcessNumber....933
2.21.1.26.4 SignalChild....933
2.21.1.27 time.c....934
2.21.1.27.1 MacintoshTime....934
2.21.1.28 total.c....935
2.21.1.28.1 TotalVehicleFuel....935
2.21.1.29 truck.c....935
2.21.1.29.1 GetTruckParameters....935
2.21.1.30 vehsubsys.c....936
2.21.1.30.1 fill_bytes....936
2.21.1.30.2 veh_subsys_encode_ground_failures....936
2.21.1.30.3 veh_subsys_encode_air_failures....937
2.21.1.31 widebdry.c....938
2.21.1.31.1 wide_bdry....938
2.21.1.32 MCC_errs.err....939
2.21.1.33 MCC_errs.c....939
2.21.1.33.1 MCC_errinit....939
2.21.1.33.2 mccerrinit_....939
2.21.2 libipc....940
2.21.2.1 ipc.h....940
2.21.2.2 keys.h....940
2.21.2.3 alarm.h....940
2.21.2.4 alarm.c....940
2.21.2.4.1 InitAlarms....941
2.21.2.4.2 SetAlarm....941
2.21.2.4.3 CancelAlarm....941
2.21.2.4.4 AlarmsEnabled....942
2.21.2.4.5 DispatchAlarm....942
2.21.2.5 error.c....943
2.21.2.5.1 IPC_SystemError....943
2.21.2.6 lock.c....944
2.21.2.6.1 AttachLocks....944
2.21.2.6.2 ResetLocks....945
2.21.2.6.3 Lock....945
2.21.2.6.4 Unlock....946
2.21.2.6.5 RemoveLocks....946
2.21.2.7 memory.c....947
2.21.2.7.1 AttachSharedMem....947
2.21.2.7.2 RemoveSharedMem....948

2.21.2.8 msgqueue.c 948
2.21.2.8.1 AttachMsgQueue 949
2.21.2.8.2 RemoveMsgQueue 949
2.21.2.9 yumm.h 950
2.21.2.10 yummextern.c 950
2.21.2.10.1 YUMMInit 950
2.21.2.10.2 YUMMProcess 951
2.21.2.10.3 OpenSocket 951
2.21.2.10.4 CloseSocket 952
2.21.2.10.5 YUMMCleanUp 953
2.21.2.10.6 SendMsg 953
2.21.2.10.7 SendReq 954
2.21.2.10.8 SendRsp 955
2.21.2.10.9 Transact 955
2.21.2.10.10 LkUpSocket 956
2.21.2.10.11 GetSocketName 957
2.21.2.10.12 ReportYUMMStatus 957
2.21.2.11 yumminit.h 957
2.21.2.12 yummintern.c 958
2.21.2.12.1 RcvMsg 958
2.21.2.12.2 GrabReqEntry 959
2.21.2.12.3 SendData 959
2.21.2.12.4 CleanupSockets 960
2.21.2.12.5 InitShMem 961
2.21.2.12.6 CleanupDeadSockets 961
2.21.2.12.7 GetInternProcID 962
2.21.2.12.8 SetUpSynchReq 962
2.21.2.13 libipcerrs.c 962
2.21.2.13.1 IPC_errinit 963
2.21.2.13.2 ipcerrinit 963
2.21.2.14 libipcerrs.err 963
2.21.3 libbbd 963
2.21.4 libmatrix 963
2.21.5 libmove 963
2.21.6 libshm 963
2.21.7 libtdb 964
2.21.7.1 cache.h 964
2.21.7.2 map.h 964
2.21.7.3 objects.h 964
2.21.7.4 tdb.h 964
2.21.7.5 tdb_local.h 965
2.21.7.6 terrain.h 965
2.21.7.7 cache_alloc.c 966

2.21.7.7.1 cache_init....967
2.21.7.7.2 cache_and_file_terminate....968
2.21.7.8 cache_cntl.c....969
2.21.7.8.1 tdb_cache_enable....969
2.21.7.8.2 tdb_cache_disable....970
2.21.7.8.3 tdb_p_cache_enabled....970
2.21.7.9 cache_data.c....970
2.21.7.10 cache_init.c....971
2.21.7.10.1 init_patch_indices....971
2.21.7.10.2 init_terrain_cache....972
2.21.7.10.3 init_cache_map....972
2.21.7.10.4 init_patch_guards....973
2.21.7.11 cache_query.c....973
2.21.7.11.1 terrain_cache_inquire....973
2.21.7.11.2 tdb_get_stripe....974
2.21.7.12 cache_queue.c....975
2.21.7.12.1 dequeue_terrain_patch....976
2.21.7.12.2 enqueue_terrain_patch....976
2.21.7.12.3 rotate_queue....976
2.21.7.13 consistent.c....977
2.21.7.13.1 tdb_consistent....977
2.21.7.14 data.c....978
2.21.7.15 dump.c....978
2.21.7.15.1 tdb_set_dumpfile....978
2.21.7.15.2 tdb_get_dumpfile....979
2.21.7.15.3 tdb_dump_terrain....979
2.21.7.15.4 dump_terrain....980
2.21.7.15.5 tdb_print_polygon....981
2.21.7.15.6 tdb_print_edge....982
2.21.7.15.7 tdb_print_object....982
2.21.7.15.8 tdb_print_trline....983
2.21.7.15.9 print_trl_hdr....984
2.21.7.15.10 tdb_print_tree....984
2.21.7.15.11 print_grid_locator....984
2.21.7.15.12 tdb_print_db_info....985
2.21.7.15.13 print_terrain_map....985
2.21.7.15.14 tdb_print_canopy....986
2.21.7.15.15 print_c_hdr....986
2.21.7.15.16 print_c_poly....987
2.21.7.15.17 tdb_print_cache_status....988
2.21.7.15.18 print_statistics....988
2.21.7.15.19 tdb_get_db_name....988
2.21.7.15.20 tdb_get_db_version....989

2.21.7.15.21 init_object_and_texture_names 989
2.21.7.15.22 get_texture_name_list 989
2.21.7.15.23 get_object_name_list 990
2.21.7.16 elevation.c 990
2.21.7.16.1 tdb_shade_get_z 990
2.21.7.16.2 tdb_get_z 992
2.21.7.16.3 find_support 993
2.21.7.16.4 p_poly_provides_support 994
2.21.7.16.5 find_height_on_poly 995
2.21.7.16.6 get_grid_number 996
2.21.7.17 error.c 996
2.21.7.17.1 tdb_error 997
2.21.7.17.2 tdb_p_on_database 997
2.21.7.18 get_patch.c 997
2.21.7.18.1 tdb_get_terrain 998
2.21.7.19 gr_loc_num.c 999
2.21.7.19.1 tdb_get_grid_number 999
2.21.7.20 h_to_w.c 1000
2.21.7.20.1 tdb_get_hull_to_world 1000
2.21.7.20.2 tracks_set_support_plane 1002
2.21.7.20.3 tracks_calc_unit_normal 1002
2.21.7.20.4 tdb_shade_place_vehicle 1003
2.21.7.20.5 tdb_place_vehicle 1004
2.21.7.21 header.c 1005
2.21.7.21.1 tdb_read_header 1005
2.21.7.22 include.c 1005
2.21.7.22.1 polygon_include 1005
2.21.7.22.2 object_include 1006
2.21.7.23 lock.c 1007
2.21.7.23.1 tdb_lock_patch 1007
2.21.7.23.2 tdb_unlock_patch 1008
2.21.7.24 map.c 1008
2.21.7.24.1 tdb_giv_utm_get_xy 1009
2.21.7.24.2 tdb_map_utm_to_xy 1009
2.21.7.24.3 tdb_giv_xy_get_utm 1010
2.21.7.24.4 tdb_map_xy_to_utm 1011
2.21.7.25 memory.c 1012
2.21.7.25.1 tdb_init_memory 1012
2.21.7.25.2 memory_init 1012
2.21.7.25.3 memory_terminate 1013
2.21.7.25.4 terrain_memory_inquire 1014
2.21.7.26 objects.c 1014
2.21.7.26.1 count_objects_in_patch 1014

2.21.7.26.2 tdb_object_count 1015
2.21.7.26.3 get_nth_object_in_patch..... 1016
2.21.7.26.4 tdb_nth_object..... 1017
2.21.7.26.5 get_closest_object_in_patch..... 1019
2.21.7.26.6 tdb_close_object..... 1020
2.21.7.26.7 get_obstr_object_in_patch..... 1021
2.21.7.26.8 tdb_obstr_object..... 1022
2.21.7.26.9 rectangle_intersected..... 1024
2.21.7.26.10 object_intersected..... 1025
2.21.7.27 tdb_init.c..... 1026
2.21.7.27.1 tdb_init_cache..... 1026
2.21.7.27.2 tdb_terminate..... 1027
2.21.7.27.3 tdb_get_tdb_info..... 1028
2.21.7.28 things.c..... 1028
2.21.7.28.1 tdb_close_thing..... 1028
2.21.7.28.2 tdb_thing_string..... 1030
2.21.7.29 treelines.c..... 1030
2.21.7.29.1 count_treelines_in_patch..... 1030
2.21.7.29.2 tdb_trline_count..... 1031
2.21.7.29.3 get_nth_treeline_in_patch..... 1032
2.21.7.29.4 tdb_nth_trline..... 1033
2.21.7.29.5 get_closest_treeline_in_patch..... 1034
2.21.7.29.6 tdb_close_trline..... 1036
2.21.7.30 trees.c..... 1037
2.21.7.30.1 count_trees_in_patch..... 1037
2.21.7.30.2 tdb_tree_count..... 1038
2.21.7.30.3 get_nth_tree_in_patch..... 1039
2.21.7.30.4 tdb_nth_tree..... 1039
2.21.7.30.5 get_closest_tree_in_patch..... 1041
2.21.7.30.6 tdb_close_tree..... 1042
2.21.7.31 version.c..... 1043
2.21.7.31.1 tdb_print_version..... 1043
2.21.7.31.2 tdb_print_format_compatible..... 1043
2.21.7.31.3 tdb_print_db_format..... 1044
2.21.7.31.4 tdb_right_format..... 1044
2.21.7.31.5 tdb_get_db_format..... 1045
2.22 THE MACINTOSH LIBRARIES AND HEADERS..... 1046
2.22.1 libmac..... 1046
2.22.1.1 libmac.h..... 1046
2.22.1.2 draw.h..... 1046
2.22.1.3 atalk.c..... 1047
2.22.1.3.1 SetUpAppleTalk..... 1047
2.22.1.3.2 SetUpATPRequest..... 1048

2.22.1.3.3 ATPPut 1049
2.22.1.3.4 DownloadTerrainMap 1049
2.22.1.3.5 NetworkEventHandler 1050
2.22.1.3.6 ATPError 1051
2.22.1.4 caution.c 1051
2.22.1.4.1 ShowCaution 1051
2.21.1.4.2 SimpleCautionEvent 1052
2.21.1.4.3 LastCaution 1053
2.22.1.4.4 CountCaution 1053
2.22.1.5 clock.h 1054
2.22.1.6 clock.c 1054
2.22.1.6.1 DrawClock 1054
2.22.1.6.2 UpdateClock 1055
2.22.1.6.3 RedrawClock 1055
2.22.1.6.4 InstallClock 1055
2.22.1.6.5 SetClock 1056
2.22.1.7 control.c 1056
2.22.1.7.1 DisableControl 1056
2.22.1.7.2 EnableControl 1057
2.22.1.8 bigprob.c 1058
2.22.1.8.1 SystemFailure 1058
2.22.1.8.2 BigProb 1058
2.22.1.9 dialog.h 1059
2.22.1.10 dialine.c 1059
2.22.1.10.1 DrawLine 1059
2.22.1.11 dialog.c 1060
2.22.1.11.1 ShowDialog 1060
2.22.1.11.2 UpdateDialog 1061
2.22.1.11.3 ThrowDialog 1062
2.22.1.11.4 DialogEvent 1062
2.22.1.11.5 CheckLastField 1063
2.22.1.11.6 SetRadioButton 1064
2.22.1.11.7 SetCheckBox 1065
2.22.1.11.8 SetText 1065
2.22.1.12 dialookup.c 1066
2.22.1.12.1 DialogLookupField 1066
2.22.1.13 diamand.c 1067
2.22.1.13.1 CheckMandatoryFields 1067
2.22.1.14 dtg.h 1068
2.22.1.15 dtg.c 1068
2.22.1.15.1 DTGToString 1068
2.22.1.15.2 StringToDTG 1068
2.22.1.15.3 DTGElapsed 1069

2.22.1.15.4 Get_DTG 1070
2.22.1.16 force.c 1070
2.22.1.16.1 LabelForceButtons 1070
2.22.1.16.2 ForceNamePStr 1071
2.22.1.17 help.h 1071
2.22.1.18 helplocal.h 1071
2.22.1.19 help.c 1071
2.22.1.19.1 helpinit 1072
2.22.1.19.2 helpshutdown 1073
2.22.1.19.3 sethelpvar 1074
2.22.1.19.4 showhelp 1074
2.22.1.19.5 centerdialog 1075
2.22.1.19.6 hpickevents 1076
2.22.1.19.7 drawlist 1077
2.22.1.19.8 setuphpick 1078
2.22.1.19.9 redrawhelp 1079
2.22.1.19.10 adjustchunks 1080
2.22.1.19.11 scrollto 1081
2.22.1.19.12 trytoscroll 1082
2.22.1.19.13 helpmove 1083
2.22.1.19.14 htopicevents 1083
2.22.1.19.15 drawtopic 1084
2.22.1.19.16 setuphtopic 1085
2.22.1.19.17 disposetopic 1086
2.22.1.19.18 helpidindex 1087
2.22.1.19.19 gethelp 1087
2.22.1.19.20 getcontenthdl 1088
2.22.1.19.21 replacevars 1089
2.22.1.19.22 createte 1090
2.22.1.19.23 createhpict 1092
2.22.1.20 init.c 1093
2.22.1.20.1 InitToolbox 1093
2.22.1.21 install.c 1093
2.22.1.21.1 InstallScrollTableEntry 1093
2.22.1.22 longpt.h 1094
2.22.1.23 longpt.c 1094
2.22.1.23.1 isqrt 1094
2.22.1.23.2 DistBetween2Pts 1095
2.22.1.23.3 PercentPt 1096
2.22.1.23.4 InterpolatePoints 1096
2.22.1.23.5 SetLongPt 1096
2.22.1.24 lookup.c 1097
2.22.1.24.1 LookupScrollTableEntry 1097

2.22.1.25 map.h 1097
2.22.1.26 map.c 1098
2.22.1.26.1 StringToMapCoordinates 1098
2.22.1.26.2 PointToMapCoordinates 1099
2.22.1.27 outline.c 1099
2.22.1.27.1 DrawItemOutline 1100
2.22.1.27.2 OutlineItem 1100
2.22.1.28 remove.c 1101
2.22.1.28.1 RemoveScrollTableEntry 1101
2.22.1.28.2 EmptyScrollTable 1102
2.22.1.29 scroll.h 1102
2.22.1.30 scroll.c 1103
2.22.1.30.1 ShowScrollTable 1103
2.22.1.30.2 DisposeScrollTable 1104
2.22.1.31 scrolldraw.c 1104
2.22.1.31.1 UpdateScrollTableEntry 1105
2.22.1.31.2 DrawScrollTable 1105
2.22.1.31.3 DrawScrollTableC 1106
2.22.1.31.4 HiliteScrollTableSelection 1107
2.22.1.32 scrollevt.c 1108
2.22.1.32.1 ScrollTableActivate 1108
2.22.1.32.2 ScrollTableEvent 1109
2.22.1.32.3 ScrollTableScrollBarEvent 1109
2.22.1.32.4 ScrollTableUp 1110
2.22.1.32.5 ScrollTableDown 1110
2.22.1.32.6 ScrollTablePage 1111
2.22.1.32.7 ScrollTableBoxScroll 1111
2.22.1.33 scrollmap.c 1112
2.22.1.33.1 MapScrollTableSelected 1112
2.22.1.34 scrollrow.c 1113
2.22.1.34.1 ScrollTableRowToEntry 1113
2.22.1.34.2 ScrollTableEntryToRow 1113
2.22.1.34.3 ScrollTableRowRect 1114
2.22.1.35 scrollto.c 1114
2.22.1.35.1 ScrollToShow 1114
2.22.1.36 scrollzoom.c 1115
2.22.1.36.1 ZoomScrollTableEntry 1115
2.22.1.37 select.c 1116
2.22.1.37.1 SelectScrollTableEntry 1116
2.22.1.37.2 SetScrollTableSelection 1117
2.22.1.37.3 CancelScrollTableSelection 1118
2.22.1.37.4 InvertScrollTableRow 1118
2.22.1.37.5 FirstScrollTableSelection 1119

2.22.1.37.6 RemoveScrollTableSelected 1120
2.22.1.38 sequence.h 1120
2.22.1.39 sequence.c 1120
2.22.1.39.1 StartDialogSeq 1121
2.22.1.39.2 ShowDialogSeqNode 1121
2.22.1.39.3 DialogSeqEvent 1122
2.22.1.39.4 DialogSeqNext 1123
2.22.1.39.5 DialogSeqPrev 1123
2.22.1.40 simple.c 1124
2.22.1.40.1 ShowSimpleDialog 1124
2.22.1.40.2 SimpleDialogEvent 1124
2.22.1.41 table.h 1125
2.22.1.42 table.c 1125
2.22.1.42.1 ShowFixTable 1125
2.22.1.42.2 UpdateFixTableRow 1126
2.22.1.42.3 FixTableEvent 1126
2.22.1.42.4 DrawFixTable 1127
2.22.1.42.5 FixTableRowRect 1129
2.22.1.43 title.c 1129
2.22.1.43.1 MenuBarTitle 1129
2.22.1.44 version.c 1130
2.22.1.44.1 ShowVersions 1130
2.22.1.44.2 AppendString 1131
2.22.1.45 wait.c 1131
2.22.1.45.1 ShowWait 1131
2.22.1.45.2 ThrowWait 1132
2.22.1.46 window.c 1132
2.22.1.46.1 SetMenuHandler 1132
2.22.1.46.2 WindowEvent 1133
2.22.1.47 zoom.c 1133
2.22.1.47.1 ZoomInit 1134
2.22.1.47.2 ZoomRect 1134
2.22.1.47.3 Blend 1135
2.22.1.47.4 ZoomToWindow 1136
2.22.2 libsim 1136
2.22.2.1 libsim.h 1136
2.22.2.2 libsim_int.h 1137
2.22.2.3 atalk.c 1138
2.22.2.3.1 DownloadSimulators 1138
2.22.2.3.2 UploadVehicleParameters 1138
2.22.2.3.3 DownloadVehicleParameters 1139
2.22.2.3.4 ProcessSimPlacedRequest 1140
2.22.2.4 data.c 1140

2.22.2.5 fred.c 1142
2.22.2.5.1 FREDSetDefaults 1142
2.22.2.5.2 FREDLoadSavedData 1142
2.22.2.5.3 FREDPlaceEvent 1143
2.22.2.6 load.c 1143
2.22.2.6.1 LoadCannedSimulators 1143
2.22.2.7 m1.c 1144
2.22.2.7.1 M1SetDefaults 1144
2.22.2.7.2 M1LoadSavedData 1145
2.22.2.7.3 M1FillDetail 1145
2.22.2.7.4 M1PlaceFetch 1146
2.22.2.7.5 M1PlaceEvent 1146
2.22.2.7.6 M1DetailEvent 1146
2.22.2.8 m2.c 1147
2.22.2.8.1 LimitCheck 1148
2.22.2.8.2 M2SetDefaults 1148
2.22.2.8.3 M2LoadSavedData 1148
2.22.2.8.4 M2FillDetail 1149
2.22.2.8.5 M2PlaceFetch 1149
2.22.2.8.6 M2PlaceEvent 1150
2.22.2.8.7 M2DetailFetch 1150
2.22.2.8.8 M2DetailEvent 1151
2.22.2.9 sim.c 1151
2.22.2.9.1 SetUpSimulators 1152
2.22.2.9.2 SimulatorTypeCStr 1152
2.22.2.9.3 SimulatorTypePStr 1152
2.22.2.9.4 CompanyName 1153
2.22.2.9.5 ShowVehicleDialog 1153
2.22.2.9.6 CompleteVehicleDialog 1154
2.22.2.9.7 ValidBumperNumber 1155
2.22.2.9.8 LocationInExerciseArea 1155
2.22.2.9.9 FindOldSimulator 1156
2.22.2.9.10 NextDefaultSlot 1157
2.22.2.10 table.c 1157
2.22.2.10.1 SimTableSetup 1157
2.22.2.10.2 SimTableUpdate 1158
2.22.2.10.3 SimTableEntry 1158
2.22.2.10.4 SimDrawBumper 1159
2.22.2.10.5 SimDrawCompany 1159
2.22.2.10.6 SimDrawLocation 1160
2.22.2.10.7 SimDrawPlaced 1160
2.22.2.10.8 SimDrawSide 1161
2.22.2.10.9 SimDrawSimulator 1162

2.22.2.10.10 SimDrawType 1162
2.22.2.11 Sim Pictures 1163
2.22.2.12 resource.h 1163
2.22.3 libsupply 1163
2.22.3.1 libsupply.h 1163
2.22.3.2 data.c 1164
2.22.3.3 version.h 1164
2.22.3.4 transfer.c 1164
2.22.3.4.1 ShowTransferAmmoDialog 1165
2.22.3.4.2 TransferAmmoFetch 1165
2.22.3.4.3 TransferAmmoEvent 1166
2.22.3.4.4 UpdateTransferAmmoLoads 1167
2.22.3.4.5 UpdateCapacityDisplay 1168
2.22.3.5 ammolist.c 1168
2.22.3.5.1 FillAmmunitionList 1169
2.22.3.5.2 NewAmmoListEntry 1169
2.22.3.5.3 InstallAmmoEntry 1170
2.22.3.5.4 RemoveAmmoEntry 1170
2.22.3.5.5 SelectAmmoEntry 1171
2.22.3.5.6 AmmoDrawName 1172
2.22.3.5.7 AmmoDrawNumber 1172
2.22.3.5.8 AmmoDrawQuantity 1173
2.22.3.5.9 AmmoDrawWeight 1174
2.22.3.5.10 AmmoDrawVolume 1174
2.22.3.5.11 TransferAmmoBetweenLists 1175
2.22.3.5.12 UpdateAmmoList 1175
2.22.3.6 libsupply_int.h 1176
2.22.3.7 summary.c 1176
2.22.3.7.1 DisplayAmmoLoadSummary 1176
2.22.3.8 supply_version.h 1177
2.22.3.9 supply.c 1177
2.22.3.9.1 TotalAmmoCapacity 1177
2.22.4 Top-level Includes 1178
2.22.4.1 basic.h 1178
2.22.4.2 battles.h 1178
2.22.4.3 repair.h 1178
2.22.4.4 obj_type.h 1178
2.22.4.5 veh_role.h 1178
2.22.4.6 veh_type.h 1178
2.22.4.7 address.h 1178
2.22.4.6 repair_m1.h 1178
2.22.4.6 repair_m2.h 1179

2.22.5 MCC Includes 1179
2.22.5.1 ammo.h 1179
2.22.5.2 arty.h 1179
2.22.5.3 console.h 1179
2.22.5.4 MCC_limits.h 1180
2.22.5.5 options.h 1180
2.22.5.6 manifest.h 1180
2.22.5.7 sim_xact.h 1180
2.22.5.8 fleet_type.h 1180
2.22.5.9 resupply.h 1180
2.22.5.10 veh_assign.h 1180

INDEX BY SECTION NUMBER INDEX-1

1 INTRODUCTION: MCC CSCI DESCRIPTION

1.1 Background

As the distributed simulation concept developed from several vehicle simulators to several dozen vehicle simulators it became evident that a method for including support units had to be developed. In addition, as the exercises derived for the many vehicle simulators increased in complexity it was recognized that a centralized method for initializing these vehicles would be useful.

The purpose of the Management, Command, and Control System (MCC) is to fulfill the dual role of providing support units to the maneuver force and assisting in exercise set-up.

Each support element is represented by a support console. The support console provides a graphical user interface to the individual responsible for coordinating the activity between the support element and the maneuver force. Support elements that are represented include logistics, maintainance, fire support and close air support. For example the Air Liason Officer uses the Close Air Support console, and the Fire Support Officer uses the Fire Support console. The console operators become integral to the exercise and are participants just as the vehicle crew members are participants.

The MCC (Management, Command and Control) CSCI is used in setting up simulations and simulating elements of the battlefield environment. It keeps track of the condition of the various manned simulators for which it is responsible as well as vehicles that it generates for service and repair. It also handles various aspects of mine laying. In addition, it monitors the available fuel and ammunition.

1.2 External Interfaces

The MCC communicates to other network entities across the SIMNET network. After an exercise is developed, the MCC is used to initialize the position and status of the individual vehicle simulators. This is accomplished using the Activate Request Protocol Data Unit (PDU). When a vehicle simulator receives an Activate Request PDU it uses the location information to place itself on the terrain and uses the status information to initialize internal status.

The majority of the network activity that the MCC participates in is concerned with the Automated Vehicles that the MCC simulates. An automated vehicle participates in network events (i.e. Collision, Impact) but is not required to traverse the terrain. These simulations are commenced when a support element (i.e. artillary piece for fire support) is requested.

The automated vehicles appear as stationary vehicles in the three dimensional vision blocks of the vehicle simulators. This automated vehicle concept increases the realism of the exercise by providing more depth to the simulated environment. The reduced complexity of the automated vehicle model allows multiple models to be simulated by a single processor.

The MCC CSCI communicates with the other CSCIs over SIMNET using Protocol Data Units (PDUs). The MCC looks for (subscribes to) specific PDUs sent out by the other CSCIs and broadcasts PDUs of its own. This communication is handled by the Mother process described in Section 2.1. Below is Table 1.2-1 which list the PDUs that are of importance to the MCC.

PDU Protocol Data Unit	CSCI Received From	CSCI Sent To	In Response To	Protocol Type	Transaction Type
Activate Request		Vehicle	Vehicle needs to be activated	Simulation	Transaction
Activate Response	Vehicle		Activate Request	Simulation	Transaction
Deactivate Request	Vehicle	Vehicle	Vehicle needs to be deactivated	Simulation	Transaction or Datagram
Deactivate Response	Vehicle		Deactivate Request	Simulation	Transaction
Vehicle Appearance		Multicast Group	Update or change vehicle appearance	Simulation	Datagram
Fire	Vehicle	Multicast Group	Directed fire	Simulation	Datagram
Impact	Vehicle		Impact of direct fire from firing vehicle	Simulation	Transaction or Datagram
Indirect Fire	Vehicle	Multicast Group	Non-direct fire	Simulation	Datagram
Collision	Vehicle		Vehicle collision	Simulation	Transaction or Datagram
Service Request	Vehicle		Vehicle needs fuel or ammo	Simulation	Datagram
Resupply Offer		Vehicle		Simulation	Datagram
Resupply Received	Vehicle			Simulation	Datagram
Resupply Cancel	Vehicle	Vehicle	Vehicle no longer meets conditions for resupply	Simulation	Datagram
Repair Request		Vehicle		Simulation	Transaction
Marker		*		Simulation	Datagram
Breached Lane Status		*		Simulation	Datagram
Minefield		*		Simulation	Datagram
Status Query	*			Data Collection	Transaction or Datagram
Status Response		*		Data Collection	Transaction or Datagram
Status Change		*		Data Collection	
Vehicle Status	Vehicle			Data Collection	Transaction or Datagram
Simulation Status		*		Data Collection	Transaction or Datagram
Error Report	Not currently used				

Table 1.2-1: PDUs used by the MCC.

1.3 Internal Structure

The MCC is composed of several processors. The physical interfaces are depicted in figure 1.3-1. The MCC host communicates to a bridge console via an RS-232 interface. Thebridge routes the communication between the Appletalk network containing the various support consoles and the RS-232 interface to the MCC host.

The processes that compose the MCC are shown in figure 1.3-2. The MCC host maintains a central process called the mother process that is the first to be activated when the MCC application is run. The mother process initializes the MCC host software and handles the activity relative to the SIMNET network. Each support console has a process executing locally on the Macintosh that handles the user interface. For each console that is active, a peer process is activated on the MCC host. Communication is directed between peers by the mother process. This architecture allows the consoles to be run independent of one another. The communication between these processes is initiated when: the support console must communicate with the SIMNET network; floating point calculations are to be made; large amounts of data must be manipulated; or when multiple consoles participate in an interaction.

The MCC CSCI is composed of the following 22 top-level CSCs':

Mother Process
SCC Process
Place Process
Admin Process
Maint Process
FSE Process
CAS Process
CEW Process
Terminal Process
Masscomp Communication Software
Bridge Process
Appletalk Software
SCC Console
Place Console
Admin Console
Maint Console
FSE Console
CAS Console
CEW Console
Network Communication Libraries
MCC Libraries
Macintosh Libraries

Figure 1.3-3 Depicts the Architecture of this CSCI.

The processes which run on the Masscomp® 5600 workstation communicate among themselves using interprocess messages. There are tables describing these messages in the appropriate CSC description. The processes on the Masscomp® 5600 workstation communicate with their corresponding applications on the Macintosh™ with a communication network consisting of an RS-232 cable, the Bridge console and an Appletalk® network. This is fully described in "The SIMNET Management, Command and Control System" document mentioned in Section 1.1.

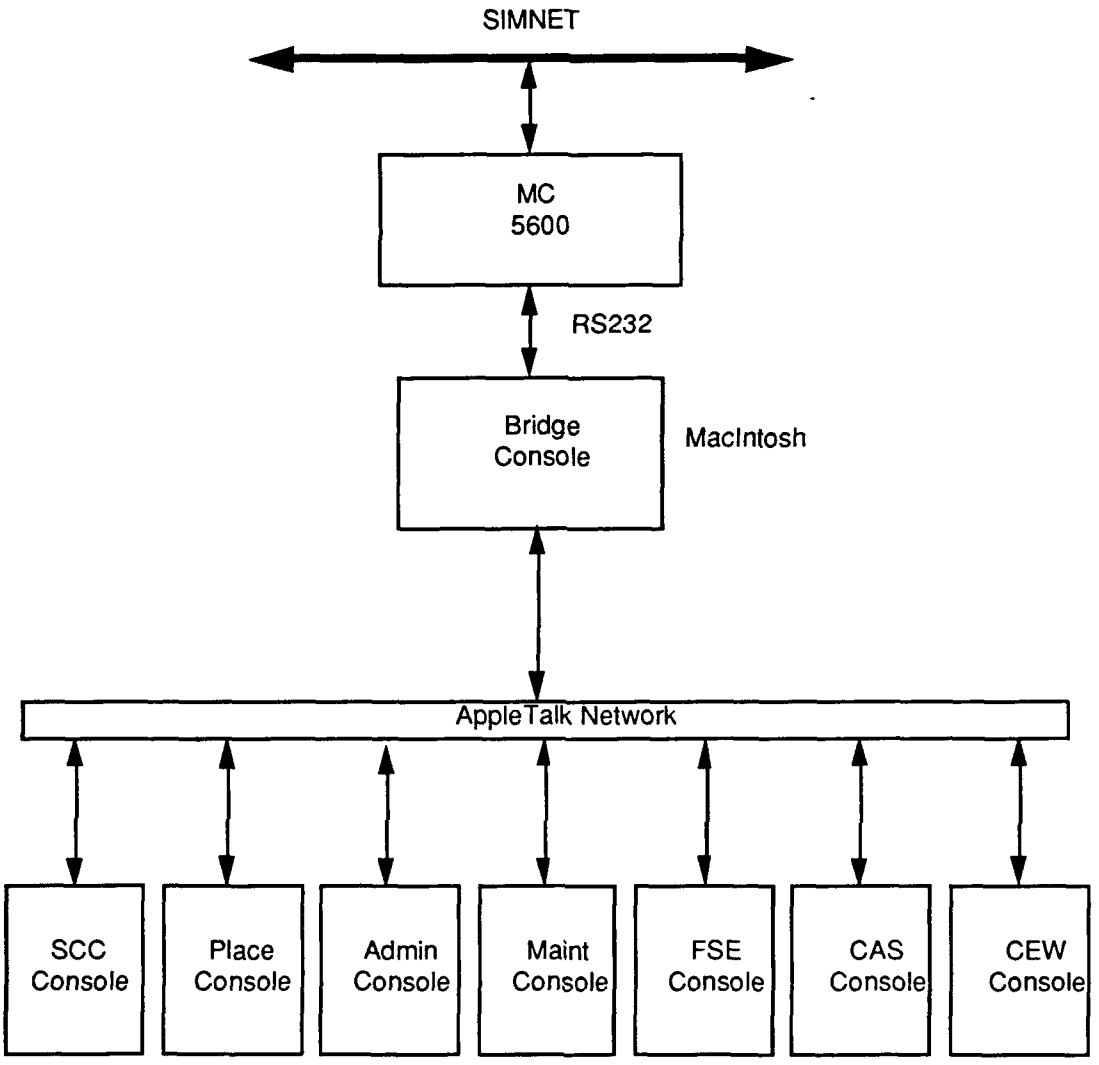


Figure 1.3-1: Internal Physical Interface.

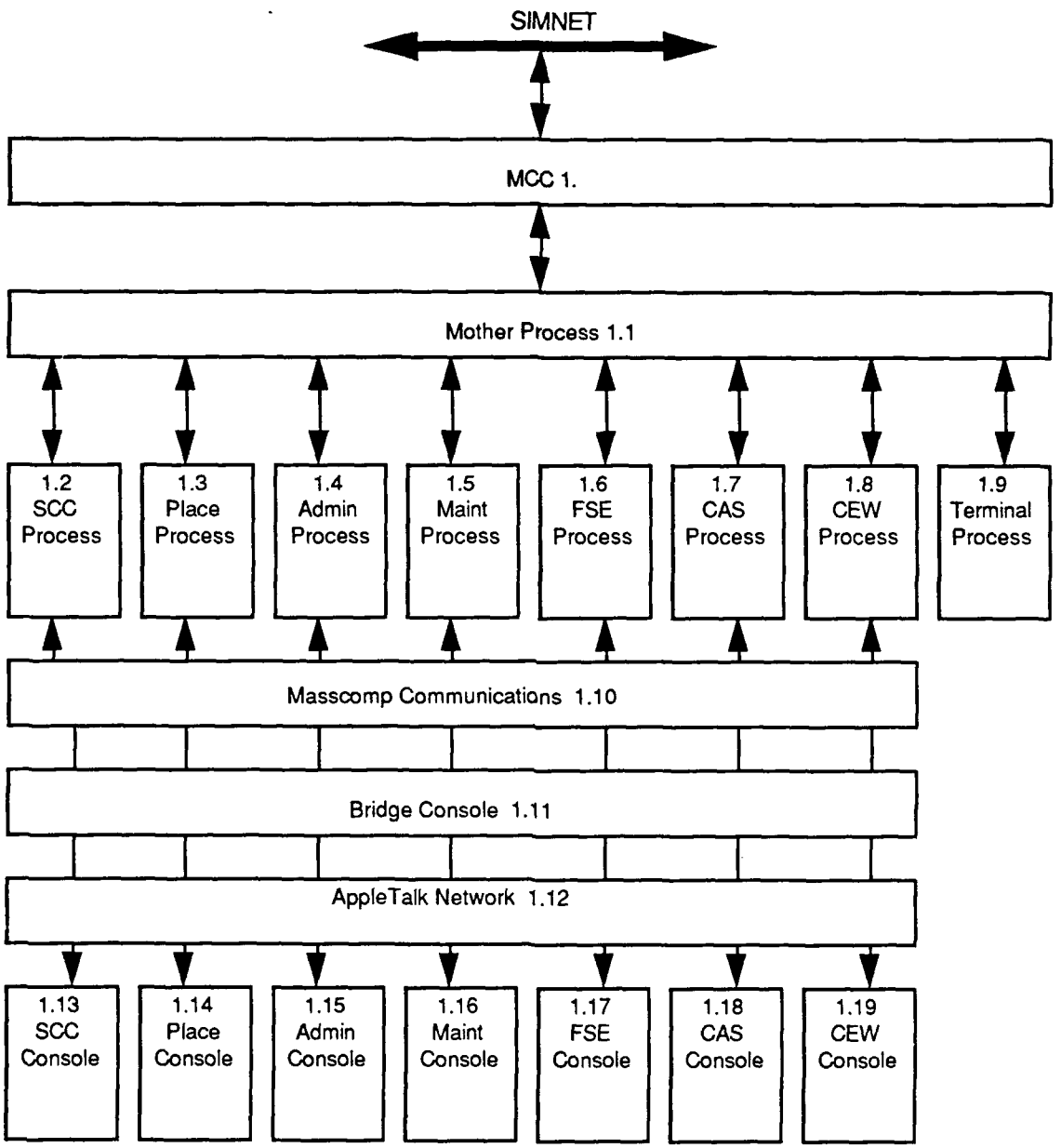


Figure 1.3-2: Internal Software Interface.

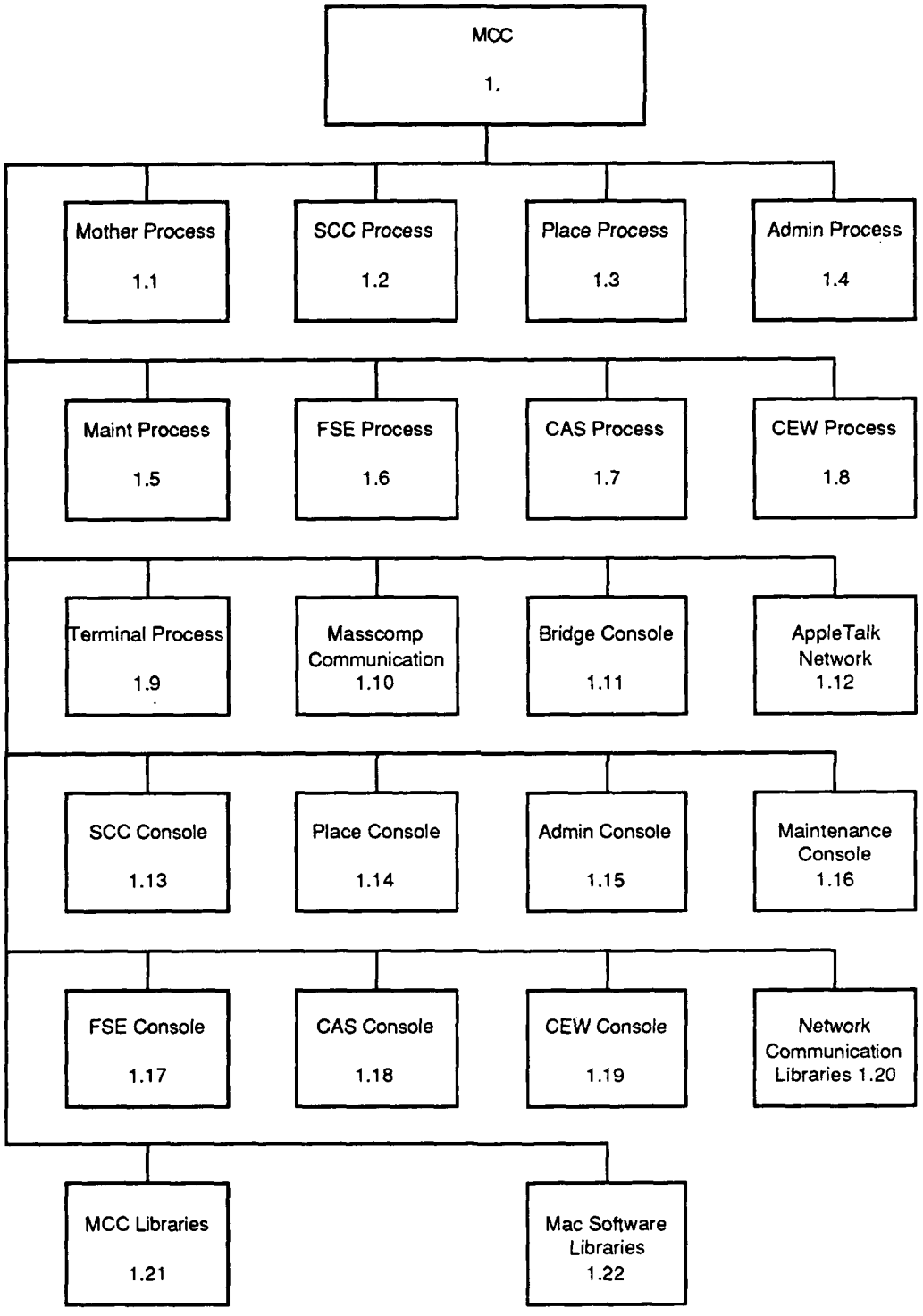


Figure 1.3-3: MCC (Management, Command and Control) CSCI Structure.

1.4 Configuration and Configuration Management

The software for the MCC is written for the most part in C. The software for the CEW Console is written in C++. The software runs on a Masscomp® 5600 workstation and eight Apple® Macintosh™ personal computers.

SIMNET software for the Macintosh is organized as a hierarchy of folders. Most software is written in C and compiled using the THINK C V4.0 compiler by Symantec. Some developer-created files contain not C source code, but rather Macintosh "resources" that describe the characteristics of a user interface, such as the layout of dialog boxes. These "resource files" are created and maintained using the Resource Editor, a developers' tool produced by Apple. The SIMNET Macintosh software is being distributed on three Macintosh 800K diskettes. The C source code files on those diskettes can be examined using any Macintosh text editor or word processor. To examine the resource files, one must use the Apple Resource Editor.

As described in the report, "The SIMNET Management, Command and Control System", each MCC console is implemented as a specialized Macintosh application program. Each of these six programs may be compiled in either of two forms: as a standalone, demo program; or as a program that communicates with an MCC host computer. Conditional compilation statements, based on a constant called "VERSION", determine which of these two versions is produced.

1.5 Terminology and Documentation

The following documents provide additional information about the MCC system and system-wide communication:

BBN Report No. 6473 - The SIMNET Management, Command and Control System
BBN Report No. 7102 - The SIMNET Network and Protocols
Top-level Descriptions for SIMNET Software, May 1990
Apple Computer, Inc. "Inside Appletalk", 1986
Apple Computer Inc. "Inside Macintosh", Volumes 1-4. Addison-Wesley Publishing Company, Inc. Reading, Mass., 1985, 1986.

2 CSC DESCRIPTIONS

2.1 The Mother Process

The Mother process performs a wide variety of tasks, including initializing MCC host software at the start of an exercise, modeling CCVs, positioning vehicles, activating vehicle simulators, recording statistics, listening to the SIMNET local area network and monitoring the MCC system as a whole. It handles all the interprocess communication for the MCC and also handles the PDUs that are sent out and received by the MCC (except for Resupply Offer and Resupply Cancel which are sent out by the Admin process). Generally, other CSCs in the MCC communicate with each other through the Mother process. Exceptions to this may occur when the SCC sends messages to other CSCs as they are initialized, and the Terminal process also sends messages as required by commands entered on the keyboard. The Mother Process structure is shown in Figure 2.1-1.

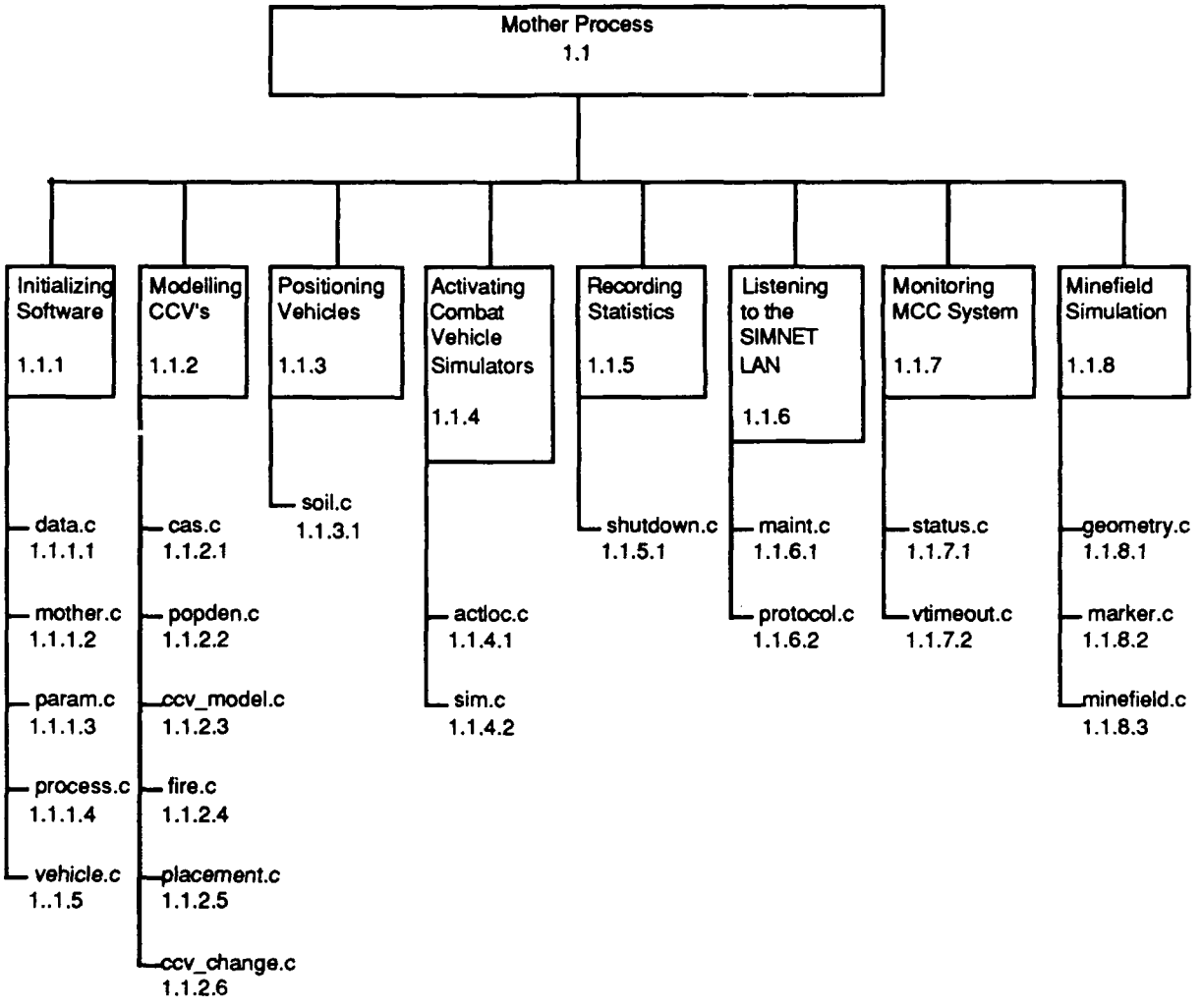


Figure 2.1-1: Mother Process Structure.

The Mother process is further broken down into eight (8) second level CSCs as follows:

Initializing Software
Modelling CCVs
Positioning Vehicles
Activating Combat Vehicle Simulators
Recording Statistics
Listening to the SIMNET LAN
Monitoring the MCC System
Minefield Simulation

2.1.1 Initializing MCC Software

This second level CSC initializes the global data structures in shared memory at start-up or whenever the exercise is recycled. It reads values in from a parameter file, /simnet/data/MCC-pars, for default values of certain variables. It also initializes the network interface and spawns the other service processes (Terminal, ATSend and ATRecv). When the MCC is started, it also spawns the SCC process which in turn starts the SCC Macintosh console. It initializes hash tables to store information received from appearance and status packets (PDUs). This CSC is composed of the following CSUs:

data.c
mother.c
param.c
process.c
vehicle.c

2.1.1.1 data.c

/simnet/mcc/Mother/data.c

data.c contains declarations of data structures used throughout the Mother process. Table 2.1-1 shows the these structures.

Variables		
Variable	**Type**	**Where Typedef Declared**
mccTermination	MCCTermination	/simnet/mcc/Mother/mother.h
exerciseLogFile	pointer to FILE	Standard C type.
vehicleLogFile	pointer to FILE	Standard C type.
howitzerShellCounter	int	Standard C type.
mortarShellCounter	int	Standard C type.
bombCounter	int	Standard C type.
smokeParametersFilename	array maxFilenameLength of char	Standard C type.

Table 2.1-1: data.c Variable Information.

2.1.1.2 mother.c

/simnet/mcc/Mother/mother.c

mother.c is the top-level module of the MCC Mother process. Table 2.1-2 describes the variables used by mother.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
networkInterface	extern int	Standard C type.
activities	array of Activity	/simnet/mcc/Mother/mother.c
PRIM	pointer to char	Standard C type.

Table 2.1-2: mother.c Variable Information.

2.1.1.2.1 ProcessMessage

ProcessMessage handles a message from another process. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the length of the data in the message received. *msg* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. *requestID* is a unique identifier which identifies a particular request which requires a response. The request ID is used when replying to a request in order to identify which request the reply is for. The function call is ProcessMessage(type, kind, length, msg, requestID). Table 2.1-3 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	register long	Standard C type.
length	int	Standard C type.
msg	register pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
requestID	long	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
alarmsEnabled	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_BAD_TERRAIN	Bad terrain database.	
MCC_UNEXPECTED_MESSAGE	Message of unexpected kind.	

Calls	
Function	**Where Described**
AlarmsEnabled	Section 2.21.2.4.4.
tdb_init_cache	Section 2.21.7.27.1.
tdb_error	Section 2.21.7.17.1.
InitPopDenMap	Section 2.1.2.2.1.
ActivateVehicle	Section 2.1.4.2.8.
DeactivateVehicle	Section 2.1.4.2.11.
PlaceCCV	Section 2.1.2.6.1.
RemoveCCV	Section 2.1.2.6.3.
PlaceBombs	Section 2.1.2.1.1.
FireVolley	Section 2.1.2.4.2.
Repair_Request	Section 2.1.6.1.3.
ResupplyOffer	Section 2.1.1.2.4.
EmplaceRequest	Section 2.1.1.2.2.
BreachRequest	Section 2.1.1.2.3.

Table 2.1-3: ProcessMessage Information.

2.1.1.2.2 EmplaceRequest

EmplaceRequest processes a message, *msg*, containing an emplacement request from the CEW. The function call is EmplaceRequest(msg). Table 2.1-4 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vertices	pointer to LongPt	/simnet/common/include/global/longpt.h
vertex_count	int	Standard C type.
Calls		
Function	**Where Described**	
wide_bdry	Section 2.21.1.31.1.	
Mines_Emplace_Minefield	Section 2.1.8.3.7.	
Marker_Emplace_Minefield	Section 2.1.8.2.1.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.1.1.2.1.	

Table 2.1-4: EmplaceRequest Information.

2.1.1.2.3 BreachRequest

BreachRequest processes a message, *msg*, containing a breach request from the CEW. The function call is BreachRequest(msg). Table 2.1-5 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
start	pointer to LongPt	/simnet/simnet/common/include/global/longpt.h
end	pointer to LongPt	/simnet/simnet/common/include/global/longpt.h
length	double	Standard C type.
bounds	array 4 of LongPt	/simnet/simnet/common/include/global/longpt.h
dx	double	Standard C type.
dy	double	Standard C type.
n_dx	double	Standard C type.
n_dy	double	Standard C type.
width	double	Standard C type.
left_x0	double	Standard C type.
left_x1	double	Standard C type.
right_x0	double	Standard C type.
right_x1	double	Standard C type.
left_y0	double	Standard C type.
left_y1	double	Standard C type.
right_y0	double	Standard C type.
right_y1	double	Standard C type.
x0	double	Standard C type.
x1	double	Standard C type.
y0	double	Standard C type.
y1	double	Standard C type.
marker_azimuth	double	Standard C type.
Calls		
Function	**Where Described**	
radians_to_simnet_angle	libapp.h	
Marker_Add_Line	Section 2.1.8.2.2.	
simnet_angle_to_radians	libapp.h	
Mines_Emplace_Breach	Section 2.1.8.3.9.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.1.1.2.1.	

Table 2.1-5: BreachRequest Information.

2.1.1.2.4 ResupplyOffer

ResupplyOffer is used to allow the Admin process to send resupply offers out to the network. Since the network channel does not robustly allow multiple concurrent accesses to the network, this function provides a mutual exclusion mechanism for preventing other network access by another entity. *msg* is a resupplyOffer PDU which is dispatched onto

the network. The function call is ResupplyOffer(msg). Table 2.1-6 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
AssocSendDatagram	Section 2.20.1.2.1.	
AssocError	Section 2.20.1.10.1.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.1.1.2.1.	

Table 2.1-6: ResupplyOffer Information.

2.1.1.2.5 PDU_Dispatch_NOP

As packets are received from the network, an index into a function table is computed using two pieces of information: 1) the type of packet, either a DATAGRAM or a TRANSACTION, and 2) the protocol group for the packet, either "simulation", "data collection", or "management" protocol. The function which processes the received packet is then called via the computed index in the function table. PDU_Dispatch_NOP is a "placeholder" routine which performs No OPeration (NOP). Initially, the entire dispatch table is set to NOP's. Then, Init_PDU_Dispatch() fills in entries in the dispatch table for packets which the MCC is interested in. PDU_Dispatch_NOP() is called in order to ignore (or not process) a received packet which does not have an initialized group/type pair. The function call is PDU_Dispatch_NOP(PDU, PDU_Len, Group, Protocol, Originator, Trans_ID, Respondent). Table 2.1-7 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	char	Standard C type.
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Originator	SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	SimulationAddress	/simnet/common/include/protocol/address.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	Always returned.

Table 2.1-7: PDU_Dispatch_NOP Information.

2.1.1.2.6 Init_PDU_Dispatch

Init_PDU_Dispatch initializes the PDU dispatch table. The function call is Init_PDU_Dispatch(). Table 2.1-8 describes the internal variables used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	unsigned int	Standard C type.
j	unsigned int	Standard C type.
ProcessSimDatagram()	extern int	Section 2.1.6.2.9.
ProcessDataColDatagram()	extern int	Section 2.1.6.2.2.
ProcessMgmtDatagram()	extern int	Section 2.1.6.2.4.
ProcessSimTransaction()	extern int	Section 2.1.6.2.6.
ProcessDataColTransaction()	extern int	Section 2.1.6.2.3.
ProcessMgmtTransaction()	extern int	Section 2.1.6.2.5.
Called By		
Function	**Where Described**	
Init_MCC_Processing	Section 2.1.1.2.8.	

Table 2.1-8: Init_PDU_Dispatch Information.

2.1.1.2.7 Check_Net

Check_Net is basically the main loop for the MCC. It runs until the MCC terminates or is restarted, reading packets off of the network. The variable *assoc_status* is returned by the call to AssocReceivePDU(). *assoc_status* is defined as follows:

-1 = an Association layer error has occurred
0 = a packet of interest was received
1 = a packet not of interest was received
2 = no packets are waiting to be received.

The function call is Check_Net(). Table 2.1-9 describes the internal variables used, errors returned, and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
assoc_status	int	Standard C type.
activity	register pointer to Activity	/simnet/mcc/Mother/mother.c
last_assoc_tick	register long	Standard C type.
PDU	pointer to char	Standard C type.
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Primitive	long	Standard C type.
Originator	SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	SimulationAddress	/simnet/common/include/protocol/address.h

Errors	
Error Name	**Reason for Error**
MCC_ASSOC_RECEIVE	Association layer error has occurred
MCC_ASSOC_TICK	Association layer error has occurred
Calls	
Function	**Where Described**
net_current_time	Section 2.20.2.8.3.
AssocReceivePDU	Section 2.20.1.6.1.
AssocError	Section 2.20.1.10.1.
AlarmsEnabled	Section 2.21.2.4.4.
AssocTickAssocLayer	Section 2.20.1.8.1.
Called By	
Function	**Where Described**
Init_MCC_Processing	Section 2.1.1.2.8.

Table 2.1-9: Check_Net Information.

2.1.1.2.8 Init_MCC_Processing

Init_MCC_Processing initializes the MCC. *process* specifies the process name of the calling process. This is used by InitializeProcess() to look up information in the process table. *param_file* is a pointer to the text string of the parameter file which should be used for MCC initialization. The function call is Init_MCC_Processing(process, param_file). Table 2.1-10 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
process	pointer to char	Standard C type.
param_file	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
now	struct timeb	
activity	pointer to Activity	/simnet/mcc/Mother/mother.c
i	int	Standard C type.

Calls	
Function	**Where Described**
YUMMInit	Section 2.21.2.10.1.
YUMMCleanup	Section 2.21.2.10.5.
Init_PDU_Dispatch	Section 2.1.1.2.6.
InitializeProcess	Section 2.21.1.19.1.
ProcessParameters	Section 2.1.1.3.19.
AlarmsEnabled	Section 2.21.2.4.4.
InitProcessTable	Section 2.1.1.4.1.
InitCCV	Section 2.1.2.3.1.
InitIndirectFire	Section 2.1.2.4.1.
InitVehicleTables	Section 2.1.1.5.2.
AssocGetSimAddress	Section 2.20.1.9.1.
net_init_time	Section 2.20.2.8.4.
ResetLocks	Section 2.21.2.6.2.
Lock	Section 2.21.2.6.3.
SpawnProcess	Section 2.21.1.26.1.
AssocSubscribe	Section 2.20.1.1.1.
MCC_SystemError	Section 2.21.1.11.3.
Act_List_Init	Section 2.1.4.1.3.
Check_Net	Section 2.1.1.2.7.
Called By	
Function	**Where Described**
main	Section 2.1.1.2.9.

Table 2.1-10: Init_MCC_Processing Information.

2.1.1.2.9 main

main is the entry point to the Mother process. The function call is main(argc, argv). Table 2.1-11 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.
Calls		
Function	**Where Described**	
set_g_ethernet_two_packets	Section 2.21.1.22.2.	
Init_MCC_Processing	Section 2.1.1.2.8.	
ResetMCC	Section 2.1.5.1.1.	
MCC_SystemError	Section 2.21.1.11.3.	

Table 2.1-11: main Information.

2.1.1.3 param.c

/simnet/mcc/Mother/param.c

param.c contains functions which process the MCC parameter file. The name of the parameter file is supplied as an argument to the command that starts up the MCC,

Init_MCC_processing, described in Section 2.1.1.2.8. Table 2.1-12 describes the variables used by param.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
kwdTable	array of struct kwd	/simnet/mcc/Mother/param.c
conTable	struct kwd	/simnet/mcc/Mother/param.c

Table 2.1-12: param.c Variable Information.

2.1.1.3.1 Battalion

Battalion reads in the Battalion parameter. The battalion entry describes a battalion number for this MCC system. Since there may be multiple MCC systems participating in a single exercise, each MCC system must have its own unique battalion number. The function call is Battalion(F). Table 2.1-13 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-13: Battalion Information.

2.1.1.3.2 Bridge

Bridge reads in the Bridge parameter. The bridge entry tells the MCC host where to find the Bridge linking it to the AppleTalk network. It has one parameter, device, which gives the name of the RS-232 port to which the Bridge is connected. The function call is Bridge(F). Table 2.1-14 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-14: Bridge Information.

2.1.1.3.3 Console

Console reads in the Console parameter. The console entry specifies the name the MCC host should use when trying to locate a console Macintosh with the AppleTalk Name Binding Protocol. The configuration file should contain a console entry for each console of the MCC system. The console entry has two parameters: type and name. Type is the type of the MCC console: scc (SIMNET Control Console), place (Vehicle Placement console), al (Admin/Log Console), maint (Maintenance Console), fse (Fire Support Console), cas (Close Air Support) or cew (Combat Engineering Workstation). Name is the name of that console's Macintosh. Because the console name may be more than one word, it must be enclosed in quotation marks. The function call is Console(F). Table 2.1-15 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
kp	pointer to struct kwd	/simnet/mcc/Mother/param.c
str	array p120 of char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
"Unknown console type"	pointer to char	Unsuccessful.
"Console name too long"	pointer to char	Unsuccessful.
"Too many placement consoles"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.
Calls		
Function	**Where Described**	
ScanString	Section 2.1.1.3.20.	

Table 2.1-15: Console Information.

2.1.1.3.3 Ex_Log

Ex_Log reads in the Ex_Log parameter. The Ex_Log entry describes a file into which the MCC system records, for each exercise: the time it entered the exercise, the duration of its involvement in the exercise, and statistics that summarize its involvement in the exercise. This entry has one parameter, file, which is the name of the file to be used by the MCC system. The function call is Ex_Log(F). Table 2.1-16 describes the parameters used and errors returned by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array 120 of char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.
Errors		
Error Name	**Reason for Error**	
MCC_OPEN_FILE_ERROR	Could not open the file.	

Table 2.1-16: Ex_Log Information.

2.1.1.3.4 Exercise

Exercise reads in the Exercise parameter. The exercise entry specifies the exercise identifier of the exercise in which this MCC system will participate. The function call is Exercise(F). Table 2.1-17 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-17: Exercise Information.

2.1.1.3.5 Password

Password reads in the Password parameter. The password entry indicates the word that the BattleMaster must enter at the SCC Console in order to perform restricted duties such as reconstituting vehicles, setting up gunnery targets, ending an exercise, etc. This entry has one parameter, secret password, which is the password to be typed in when BattleMaster functions are desired. The function call is Password(F). Table 2.1-18 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array 120 of char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
"BattleMaster password too long"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-18: Password Information.

2.1.1.3.6 Smoke

Smoke reads in the Smoke parameter. The smoke entry describes a file which has parameters to model the white phosphorous smoke clouds. This entry has one parameter, file, which is the name of the file to be used by the MCC system. The function call is Smoke(F). Table 2.1-19 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array 120 of char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
"Smoke parameters filename too long"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-19: Smoke Information.

2.1.1.3.7 Terrain

Terrain reads in the Terrain parameter. The terrain entry describes a terrain database stored on the MCC host's disk drive. The configuration file should contain one terrain entry for each terrain database available to the MCC system. The terrain entry has one parameter, terrain database, which is the name of a terrain database on the host's filesystem. The function call is Terrain(F). Table 2.1-20 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array 120 of char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Too many terrain databases"	pointer to char	Unsuccessful.
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
"Terrain filename too long"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-20: Terrain Information.

2.1.1.3.8 Veh_Log

Veh_Log reads in the Veh_Log parameter. The Veh_Log entry describes a file into which the MCC system records, for each exercise, information about the use of each simulated vehicle deployed by the MCC system. Also recorded is a unique exercise key which may be used to cross-reference data with that in the log file. This entry has one parameter, file, which is the name of the file to be used by the MCC system. The routine should return NULL if sucessful and "Missing or incorrect parameter value" when the number of parameters in the Veh_Log entry is not correct. Veh_log expects a single parameter, *F*, which is the name of the file to use as the vehicle log file. The function call is Veh_Log(F). Table 2.1-21 describes the parameters used and errors returned by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array 120 of char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	The number of parameters in the Veh_Log entry is not correct.
NULL	pointer to char	Successful.
Errors		
Error Name	**Reason for Error**	
MCC_OPEN_FILE_ERROR	Could not open the file.	

Table 2.1-21: Veh_Log Information.

2.1.1.3.9 Vehicle

Vehicle reads in the Vehicle parameter. The vehicle entry describes a vehicle simulator to the MCC system. The configuration file should contain a vehicle entry for each simulator that is available for use in the simulation exercise. The parameters of the vehicle entry are,

in order: address, trailer, trailer element and type. Address is the simulator's address on the SIMNET local area network. Trailer and trailer element specify the physical location of the simulator; they are a number and a letter, respectively. Type is the type of vehicle that the simulator simulates. The function call is Vehicle(F). Table 2.1-22 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
str	array 120 of char	Standard C type.
i	int	Standard C type.
j	int	Standard C type.
k	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Too many vehicle simulators"	pointer to char	Unsuccessful.
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-22: Vehicle Information.

2.1.1.3.10 VList

VList reads in the vlist parameter. The vlist entry specifies the size of the vehicle list. This is the maximum number of vehicles which can be in an exercise that the MCC is a part of. The size of the vehicle status list is the same. The function call is VList(F). Table 2.1-23 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-23: VList Information.

2.1.1.3.11 VehHashTableSize

VehHashTableSize reads in the vehhashsize parameter. The vehhashsize entry specifies the size of the internal hash-table used to index vehicles in the vehicle list. vehhashsize should always be a prime number. The function call is VehHashTableSize(F). Table 2.1-24 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-24: VehHashTableSize Information.

2.1.1.3.12 StatusHashSize

StatusHashSize reads in the statushashsize parameter. The statushashsize entry specifies the size of the internal hash-table used to index previously saved vehicle status packets. statushashsize should always be a prime number. The function call is StatusHashSize(F). Table 2.1-25 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-25: StatusHashSize Information.

2.1.1.3.13 VehicleMarkingR

VehicleMarkingR reads in the vehiclemarking parameter. The vehiclemarking parameter specifies a fixed string which should be prepended to the text field of the vehicleMarking field in VehicleAppearance PDUs generated by MCC ccvs. The function call is VehicleMarkingR(F). Table 2.1-26 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
NULL	pointer to char	Successful.

Table 2.1-26: VehicleMarkingR Information.

2.1.1.3.14 CurBattleScheme

CurBattleScheme reads in the battlescheme parameter. The function call is CurBattleScheme(F). Table 2.1-27 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect parameter value"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-27: CurBattleScheme Information.

2.1.1.3.15 DefineBattleScheme

DefineBattleScheme reads in the battleScheme matrices. The function call is DefineBattleScheme(F). Table 2.1-28 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
j	int	Standard C type.
scheme	pointer to BattleSchemes	/simnet/mcc/libmcc/guise.h
country	array 8 of char	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect scheme number"	pointer to char	Unsuccessful.
"Incorrect scheme number"	pointer to char	Unsuccessful.
"Missing country param"	pointer to char	Unsuccessful.
"Invalid country param"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-28: DefineBattleScheme Information.

2.1.1.3.16 MineScanInterval

MineScanInterval reads in the minescan parameter. The minescasn entry specifies how often the MCC checks for vehicles passing through minefields that the MCC is simulating. The value is in milliseconds and the MCC typically checks 1/3 of the vehicles in an exercise during each scan. The function call is MineScanInterval(F). Table 2.1-29 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect scan interval number"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-29: MineScanInterval Information.

2.1.1.3.17 MarkerSpacing

MarkerSpacing reads in the makerspacing parameter. The markerspacing entry specifies how far apart each flag which is placed around the borders of an emplaced minefield or breached lane should be from the previous one. The function call is MarkerSpacing(F). Table 2.1-30 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect marking spacing number"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-30: MarkerSpacing Information.

2.1.1.3.18 MarkerInterval

MarkerInterval reads in the markerbroadcast parameter. The markerbroadcast entry specifies how frequently, in milliseconds, the minefield marker flags are broadcast to the network. The function call is MarkerInterval(F). Table 2.1-31 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
F	pointer to FILE	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
"Missing or incorrect marker broadcast interval number"	pointer to char	Unsuccessful.
NULL	pointer to char	Successful.

Table 2.1-31: MarkerInterval Information.

2.1.1.3.19 ProcessParameters

ProcessParameters parses the MCC parameter file. ProcessParameters reports errors in the condition that:

-- The parameter file cannot be opened
-- The routine called to deal with the parameter returns a non-null text string (the error text)
-- A keyword is seen which does not match any keywords in the keyword table
-- Extraneous information is recognized on a line of data

The function call is ProcessParameters(filename). Table 2.1-32 describes the parameters used and errors returned using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
filename	pointer to char	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
f	pointer to FILE	Standard C type.
i	int	Standard C type.
line	int	Standard C type.
kp	pointer to struct kwd	/simnet/mcc/Mother/param.c
kwdString	array 50 of char	Standard C type.
errString	pointer to char	Standard C type.
str	array 120 of char	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_OPEN_FILE_ERROR	The parameter cannot be opened.	
MCC_PARAMETER_ERROR	Error in a parameter.	
Called By		
Function	**Where Described**	
Init_MCC_Processing	Section 2.1.1.2.8.	

Table 2.1-32: ProcessParameters Information.

2.1.1.3.20 ScanString

ScanString reads a string enclosed in quotes. The function call is ScanString(f, cp). Table 2.1-33 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
f	pointer to FILE	Standard C type.
cp	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	String does not have leading quote.
1	int	Successful.
Called By		
Function	**Where Described**	
Console	Section 2.1.1.3.3.	

Table 2.1-33: ScanString Information.

2.1.1.4 process.c

/simnet/mcc/Mother/process.c

process.c contains a function that initializes a table of the MCC process. Table 2.1-34 describes the variables used by process.c

Variables		
Variable	**Type**	**Where Typedef Declared**
process	array of struct process	/simnet/mcc/Mother/process.c

Table 2.1-34: process.c Variable Information.

2.1.1.4.1 InitProcessTable

InitProcessTable initializes the process table in shared memory. Note that before changing processes, we change into a subdirectory to verify that the filename to run is in the original directory. The function call is InitProcessTable(). Table 2.1-35 describes the internal variable used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Called By		
Function	**Where Described**	
Init_MCC_Processing	Section 2.1.1.2.8.	

Table 2.1-35: InitProcessTable Information.

2.1.1.5 vehicle.c
/simnet/mcc/Mother/vehicle.c

vehicle.c contains functions that manage the vehicle lists and tables.

2.1.1.5.1 AllocShMem

AllocShMem is used to allocate shared memory for vehicle and status lists. *Size* is the size in bytes of the area to allocate and *Param* is a pointer to the MCC parameter block. Information in the parameter block is used to create the shared memory segment. The function call is AllocShMem(Size, Param). Table 2.1-36 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Size	int	Standard C type.
Param	MCCParameters	/simnet/mcc/include/MCC_pars.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
memloc	pointer to char	Standard C type.
shmid	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
memloc	pointer to char	Pointer to the allocated memory area

Calls	
Function	**Where Described**
AttachSharedMem	Section 2.21.2.7.1.

Table 2.1-36: AllocShMem Information.

2.1.1.5.2 InitVehicleTables

InitVehicleTables initializes the tables of CCVs, simulators and vehicles. The function call is InitVehicleTables(). Table 2.1-37 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
Calls		
Function	**Where Described**	
FQueue_Create	Section 2.21.1.13.3.	
Called By		
Function	**Where Described**	
Init_MCC_Processing	Section 2.1.1.2.8.	

Table 2.1-37: InitVehicleTables Information.

2.1.1.5.3 VehicleIDToSim

VehicleIDToSim maps a vehicle ID, *vehicleID*, to a pointer to the SimulatorStatus for the simulator owning the ID. It returns a pointer to the SimulatorStatus, or 0 if the vehicle ID does not belong to one of our simulators. The function call is VehicleIDToSim(vehicleID). Table 2.1-38 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
Return Values		
Return Value	**Type**	**Meaning**
0	pointer to SimulatorStatus	Vehicle does not belong to one of our simulators.
(&vehicles->sim[veh->index])	pointer to SimulatorStatus	Pointer to SimulatorStatus for simulator that owns vehicle.

Calls	
Function	**Where Described**
FQueue_Retrieve	Section 2.21.1.13.7.
Called By	
Function	**Where Described**
CountRoundsFired	Section 2.1.6.2.10.
VTimeout_Appearance_Timeout	Section 2.1.7.2.1.

Table 2.1-38: VehicleIDToSim Information.

2.1.1.5.4 GetVehicleLocation

GetVehicleLocation obtains a vehicle's location. *vehicleID* is used to specify the vehicle; *location* and *vehicleGuises* are returned information about the vehicle. The lookup is performed using either the internal ccv table for the local MCC vehicles or by checking the vehicle list for appearance PDU if the vehicle is a remote vehicle. The function call is GetVehicleLocation(vehicleID, vehicleGuises, location). Table 2.1-39 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
vehicleGuises	pointer to VehicleGuises	/simnet/common/include/protocol/basic.h
location	array of float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
ccv	pointer to CCVStatus	/simnet/mcc/include/veh_table.h
VA	VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
net_current_time	Section 2.20.2.8.3.	
Called By		
Function	**Where Described**	
GuideBomber	Section 2.1.2.1.2.	
NearbyVehicle	Section 2.1.2.5.2.	
NearestCombatVehicle	Section 2.1.2.5.3.	

Table 2.1-39: GetVehicleLocation Information.

2.1.1.5.5 IsGroundVehicle

IsGroundVehicle determines whether a vehicle type, given by *vehicleGuise*, represents a ground vehicle. The function call is IsGroundVehicle(vehicleGuise). Table 2.1-40 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleGuise	ObjectType	/simnet/common/include/protocol/basic.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	Not a ground vehicle.
1	int	Ground vehicle.
Called By		
Function	**Where Described**	
GuideBomber	Section 2.1.2.1.2.	
NearbyVehicle	Section 2.1.2.5.2.	
NearestCombatVehicle	Section 2.1.2.5.3.	

Table 2.1-40: IsGroundVehicle Information.

2.1.1.5.6 IsCombatVehicle

IsCombatVehicle determines whether a vehicle type, given by *guise*, represents a combat vehicle. Only ground vehicles are checked. The function call is IsCombatVehicle(guise). Table 2.1-41 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
guise	ObjectType	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
obj_function	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Not a combat vehicle.
1	int	Combat vehicle.
Called By		
Function	**Where Described**	
NearestCombatVehicle	Section 2.1.2.5.3.	

Table 2.1-41: IsCombatVehicle Information.

2.1.2 Modeling Computer-Controlled Vehicles (CCVs)

This software initializes the CCVs according to input from the Macintosh consoles. It issues appearance packets for them periodically when they are visible and issues a deactivate packet when they should be made invisible. It notices when one of the CCVs has been hit by something on the battle field and assesses damages accordingly. It also keeps track of when indirect fire is due to be dropped and issues the appropriate PDUs for mortars, howitzers and close air support bombs. In addition, it keeps a CCV Population Density Map which maps the locations of the CCVs into 500 meter by 500 meter squares to enable faster processing when it needs to search for CCVs in a given area. This functionality is realized by the following CSUs:

cas.c
ccv_change.c
ccv_model.c
fire.c
placement.c
popden.c

2.1.2.1 cas.c

/simnet/mcc/Mother/cas.c

cas.c contains functions for processing requests to drop bombs from the CAS process. Table 2.1-42 shows the variable used by cas.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
nextBombEventID	unsigned short	Standard C type.

Table 2.1-42: cas.c Variable Information.

2.1.2.1.1 PlaceBombs

PlaceBombs lays a sortie's worth of bombs. This function passes the *location* and *heading* to GuideBomber, which changes them within casTargetRange and casHeadingRange, if necessary, to make the sortie more effective. It then computes the (x,y) for each bomb, using location, heading and bomb scatter parameters from MCC_data.h, puts these in a VolleyDescriptor along with the delay between detonations and a unique event ID for each detonation. Finally, PlaceIndirectFire (Section 2.1.2.4.4.) is called to generate Indirect Fire PDUs for the detonations. The function call is PlaceBombs(location, heading). Table 2.1-43 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
location	array of float	Standard C type.
heading	float	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
det	register pointer to struct detonation	
i	register short	Standard C type.
sx	float	Standard C type.
sy	float	Standard C type.
cx	float	Standard C type.
cy	float	Standard C type.
volley	VolleyDescriptor	/simnet/mcc/Mother/mother.h
Calls		
Function	**Where Described**	
GuideBomber	Section 2.1.2.1.2.	
PlaceIndirectFire	Section 2.1.2.4.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.1.1.2.1.	

Table 2.1-43: PlaceBombs Information.

2.1.2.1.2 GuideBomber

GuideBomber adjusts the location and heading of a CAS sortie's bombing. *targetLocation* is the location to which the sortie was dispatched and *heading* points to the run-in heading requested (in degrees clockwise from north). The function call is GuideBomber(targetLocation, heading). Table 2.1-44 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
targetLocation	array of float	Standard C type.
heading	pointer to float	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
i	register unsigned short	Standard C type.
vehLocation	array 2 of float	Standard C type.
dx	float	Standard C type.
dy	float	Standard C type.
oldHeading	float	Standard C type.
newHeading	float	Standard C type.
ccvList	array numberCCVS of unsigned short	Standard C type.
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
vlist_ptr	pointer to FQueue_t	/simnet/mcc/libmcc/fqueue.h
VA	pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
NearbyCCVs	Section 2.1.2.2.4.	
ConsiderVehicle	Section 2.1.2.1.3.	
FQueue_Scan	Section 2.21.2.13.6.	
FQueue_Retrieve	*Section 2.21.2.13.7.*	
GetVehicleLocation	Section 2.1.1.5.4.	
IsGroundVehicle	Section 2.1.1.5.5.	
deg_to_rad	Macro defined in /simnet/common/include/global/sim_macros.h.	
Called By		
Function	**Where Described**	
PlaceBombs	Section 2.1.2.1.1.	

Table 2.1-44: GuideBomber Information.

2.1.2.1.3 ConsiderVehicle

ConsiderVehicle evaluates a vehicle on the ground as a target for a CAS attack. A list of vehicles within range of the target location is accumulated in global data structures. *targetLocation* is the x, y and z coordinates of the intended target area. *vehicleID* is the ID of a vehicle which should be considered a target. *vehLocation* is th x, y and z coordinates corresponding with the ID of the vehicle that is being considered a target. This location is compared with *targetLocation* to see if it is within the tolerances used by the CAS simulation. The function call is ConsiderVehicle(targetLocation, vehicleID, vehLocation). Table 2.1-45 describes the parameters used and errors returned using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
targetLocation	array of float	Standard C type.
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
vehLocation	array of float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dx	float	Standard C type.
dy	float	Standard C type.
distance	float	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_TOO_MANY_TARGETS	Too many targets (more than maxCASTargets) for the CAS pilot to handle.	
Called By		
Function	**Where Described**	
GuideBomber	Section 2.1.2.1.2.	

Table 2.1-45: ConsiderVehicle Information.

2.1.2.2 popden.c

/simnet/mcc/Mother/popden.c

popden.c contains the routines for the population density map of computer controlled vehicles. The terrain is divided up into buckets 500 meters square. Each bucket contains a pointer (in actuality, an array index) into a doubly linked list of CCVs in that bucket. Each CCV in the list also contains the coordinate of the bucket to which it belongs. The actual storage for the list is an array of PopDenEntry elements which is indexable by the number of the CCV whose links are contained therein. Table 2.1-46 shows the variables used by popden.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
numberOfBuckets	int	Standard C type.
pdmBucketsX	short	Standard C type.
pdmBucketsY	short	Standard C type.
pdmIndex	pointer to short	Standard C type.
pdmEntry	pointer to PopDenMapEntry	/simnet/mcc/Mother/popden.c

Table 2.1-46: popden.c Variable Information.

2.1.2.2.1 InitPopDenMap

InitPopDenMap initializes the Population DensityMap for the particular size of terrain chosen for the exercise. The function call is InitPopDenMap(). Table 2.1-47 describes the internal variable used and errors returned using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_NO_MEMORY	The MCC has run out of memory.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.1.1.2.1.	

Table 2.1-47: InitPopDenMap Information.

2.1.2.2.2 PDMInsert

PDMInsert inserts an entry into the Population Density Map for a CCV that has been placed on the terrain. *vehicle* is the number of the CCV being placed and *location* is the vehicle's new location on the terrain. The function call is PDMInsert(vehicle, location). Table 2.1-48 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	unsigned short	Standard C type.
location	array of float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
bucket	register int	Standard C type.
entry	register pointer to PopDenMapEntry	/simnet/mcc/Mother/popden.c
Errors		
Error Name	**Reason for Error**	
MCC_INCONSISTENCY	A bucket in the population density map lies outside the memory area.	
Calls		
Function	**Where Described**	
PDMDelete	Section 2.1.2.2.3.	
Called By		
Function	**Where Described**	
PlaceCCV	Section 2.1.2.6.1.	

Table 2.1-48: PDMInsert Information.

2.1.2.2.3 PDMDelete

PDMDelete removes a vehicle's entry, *vehicle*, from the Population Density Map. The function call is PDMDelete(vehicle). Table 2.1-49 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	unsigned short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
bucket	register int	Standard C type.
entry	register pointer to PopDenMapEntry	/simnet/mcc/Mother/popden.c
Called By		
Function	**Where Described**	
PDMInsert	Section 2.1.2.2.2.	
BroadcastCCVBatch	Section 2.1.2.3.2.	

Table 2.1-49: PDMDelete Information.

2.1.2.2.4 NearbyCCVs

NearbyCCVs prepares a list of the CCVs near a specified location. *location* is the coordinates of the center of the area of interest. *range* is the range about the location of interest and *list* is the vector to be filled with CCV numbers. *list* must be large enough to accomodate the "worst case" of numberCCVs entries. The function call is NearbyCCVs(location, range, list). Table 2.1-50 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
location	array of float	Standard C type.
range	int	Standard C type.
list	array of unsigned short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
start_X	int	Standard C type.
start_Y	int	Standard C type.
end_X	int	Standard C type.
end_Y	int	Standard C type.
bX	int	Standard C type.
bY	int	Standard C type.
bucket	register short	Standard C type.
vehicle	register short	Standard C type.
vp	register pointer to unsigned short	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
(vp - list)	int	List of CCVs in area.

Called By	
Function	**Where Described**
GuideBomber	Section 2.1.2.1.2.
IndirectFireDamage	Section 2.1.2.4.6.
NearbyVehicle	Section 2.1.2.5.2.

Table 2.1-50: NearbyCCVs Information.

2.1.2.3 ccv_model.c

/simnet/mcc/Mother/ccv_model.c

ccv_model.c contains routines for modelling the behavior of CCVs. The MCC must broadcast VehicleAppearance PDUs periodically for all visible CCVs. For CCVs other than gunnery targets, it broadcasts PDUs in batches -- a few batches a second -- so that it cycles through all CCVs in five seconds. PDUs are broadcast for gunnery targets with turrets every second so that they will appear to be smoothly scanning their guns. When a CCV changes appearance, a PDU is always broadcast for it at the next batch. Table 2.1-51 shows the variables used by ccv_model.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
howitzerMuzzle	array 3 of float	Standard C type.
mortarMuzzle	array 3 of float	Standard C type.
batchSize	short	Standard C type.
lastBatchTime	long	Standard C type.
nextCCV	short	Standard C type.
vaBuf	SimulationPDU	/simnet/common/include/protocol/p_sim.h
daBuf	SimulationPDU	/simnet/common/include/protocol/p_sim.h
fiBuf	SimulationPDU	/simnet/common/include/protocol/p_sim.h
scBuf	DataCollectionPDU	/simnet/common/include/protocol/p_data.h
networkInterface	extern int	Standard C type.

Table 2.1-51: ccv_model.c Variable Information.

2.1.2.3.1 InitCCV

InitCCV initializes this CCV modelling package. The function call is InitCCV(). Table 2.1-52 describes the internal variables used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
batchesPerCycle	short	Standard C type.
i	short	Standard C type.
Called By		
Function	**Where Described**	
Init_MCC_Processing	Section 2.1.1.2.8.	

Table 2.1-52: InitCCV Information.

2.1.2.3.2 BroadcastCCVBatch

BroadcastCCVBatch broadcasts a batch of PDUs for some CCVs. The function call is BroadcastCCVBatch(). Table 2.1-53 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vehicle	register unsigned short	Standard C type.
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
i	unsigned short	Standard C type.
newSecond	unsigned short	Standard C type.
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Return Values		
Return Value	**Type**	**Meaning**
ccvBatchPeriod	int	Delay time before calling function again. ccvBatchPeriod is a constant defined in /simnet/mcc/include/MCC_timers.h.
Calls		
Function	**Where Described**	
SendMCCMsg2	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
AssocSendDatagram	Section 2.20.1.2.1.	
PRO_DATA_STATUS_CHANGE_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	
PDMDelete	Section 2.1.2.2.3.	
DeactivateVehicle	Section 2.1.4.2.11.	
BroadcastAppearance	Section 2.1.2.3.3.	

Table 2.1-53: BroadcastCCVBatch Information.

2.1.2.3.3 BroadcastAppearance

BroadcastAppearance broadcasts a Vehicle Appearance PDU for a CCV, *vehicle*. The function call is BroadcastAppearance(vehicle). Table 2.1-54 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	unsigned short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h

Calls	
Function	**Where Described**
AssocSendDatagram	Section 2.20.1.2.1.
PRO_SIM_APPEARANCE_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h
Called By	
Function	**Where Described**
BroadcastCCVBatch	Section 2.1.2.3.2.

Table 2.1-54: BroadcastAppearance Information.

2.1.2.3.4 BroadcastDeactivate

BroadcastDeactivate broadcasts a Deactivate PDU for either a CCV or a simulator, determined by *vehicleID*. The function call is BroadcastDeactivate(vehicleID). Table 2.1-55 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
AssocSendTransact	Section 2.20.1.4.1.	

Table 2.1-55: BroadcastDeactivate Information.

2.1.2.3.5 FireShell

FireShell fires a shell from a mortar or howitzer. *vehicle* is the ccv number of the vehicle firing the mortar/howitzer shell. *projectile* is the SIMNET projectile being fired. *detonator* is the SIMNET detonator on the projectile being fired. *eventID* is a pointer to a SIMNET eventID to use in the fire and impact packet sent on the network. Events on the network are linked via the *eventID*. The function call is FireShell(vehicle, projectile, detonator, eventID). Table 2.1-56 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	unsigned short	Standard C type.
projectile	ObjectType	/simnet/common/include/protocol/basic.h
detonator	ObjectType	/simnet/common/include/protocol/basic.h
eventID	pointer to unsigned short	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
i	register short	Standard C type.
muzzle	pointer to float	Standard C type.
temp	array 3 of float	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
0	int	Unsuccessful.
1	int	Successful. The firing vehicle is healthy and visible.

Calls	
Function	**Where Described**
fvec_mat_mul	Section 2.6.2.22.1 in the Vehicles CSCI.
AssocSendDatagram	Section 2.20.1.2.1.
PRO_SIM_FIRE_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h

Called By	
Function	**Where Described**
CheckVolleyTable	Section 2.1.2.4.3.

Table 2.1-56: FireShell Information.

2.1.2.4 fire.c

/simnet/mcc/Mother/fire.c

fire.c contains routines which support the simulation of indirect fire. Table 2.1-57 shows the variables used by fire.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
networkInterface	extern int	Standard C type.
indBuf	SimulationPDU	/simnet/common/include/protocol/p_sim.h
volleyTable	pointer to VolleyDescriptor	/simnet/mc/Mother/mother.h

Table 2.1-57: fire.c Variable Information.

2.1.2.4.1 InitIndirectFire

InitIndirectFire initializes the indirect fire package. The function call is InitIndirectFire(). Table 2.1-58 describes the internal variable used and errors returned using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.

Errors	
Error Name	**Reason for Error**
MCC_NO_MEMORY	Couldn't allocate the table of descriptions of volleys in progress.
Called By	
Function	**Where Described**
Init_MCC_Processing	Section 2.1.1.2.8.

Table 2.1-58: InitIndirectFire Information.

2.1.2.4.2 FireVolley

FireVolley queues volleys of indirect fire for firing at some later time. This call is invoked due to an IPC message from the FSE process. It uses the "volley" structure of the MCCMessageBuffer structure defined in mcc/include/MCC_ipc.h. *msg* specifies the quantity and type of fire which has been called for. The function call is FireVolley(msg). Table 2.1-59 describes the parameter used and errors returned using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
volley	register pointer to VolleyDescriptor	/simnet/mcc/Mother/mother.h
i	register short	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_NO_VOLLEY	Maximum number of valleys already queued; there is no space for more volleys at the current time.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.1.1.2.1.	

Table 2.1-59: FireVolley Information.

2.1.2.4.3 CheckVolleyTable

CheckVolleyTable cycles through the table of active volleys, looking for something that needs to be done, such as firing a shell by broadcasting a Fire PDU or reporting the impact of the shells with an Indirect Fire PDU. The function call is CheckVolleyTable(). Table 2.1-60 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
volley	register pointer to VolleyDescriptor	/simnet/mcc/Mother/mother.c
i	register short	Standard C type.
j	register short	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
volleyTablePeriod	int	Delay before calling the procedure again. volleyTablePeriod is a constant defined in /simnet/mcc/include/MCC_timers.h.
Calls		
Function	**Where Described**	
FireShell	Section 2.1.2.3.5.	
PlaceIndirectFire	Section 2.1.2.4.4.	

Table 2.1-60: CheckVolleyTable Information.

2.1.2.4.4 PlaceIndirectFire

PlaceIndirectFire generates indirect fire detonations on the terrain. *volley* describes the type, quantity and location of detonations which are to be placed on the terrain. The function call is PlaceIndirectFire(volley). Table 2.1-61 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
volley	register pointer to VolleyDescriptor	/simnet/mcc/Mother/mother.c
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
msgDet	register pointer to struct detonation	
pduDet	register pointer to IndirectFireDetonation	/simnet/common/include/protocol/p_sim.h
accumDelay	int	Standard C type.
prevDelay	int	Standard C type.
msgDetIdx	short	Standard C type.
pduDetIdx	short	Standard C type.
delays	array maxIndirectFireDetonations of unsigned char	Standard C type.
location	TDB_POINT	/simnet/common/libsrc/libtdb/tdb.h

Errors	
Error Name	**Reason for Error**
MCC_BAD_TERRAIN	Terrain location specified in the volley descriptor is not valid.
Calls	
Function	**Where Described**
SendIndirectFirePDU	Section 2.1.2.4.5.
tdb_get_z	Section 2.21.7.16.2.
tdb_error	Section 2.21.7.17.1.
Called By	
Function	**Where Described**
CheckVolleyTable	Section 2.1.2.4.3.
PlaceBombs	Section 2.1.2.1.1.

Table 2.1-61: PlaceIndirectFire Information.

2.1.2.4.5 SendIndirectFirePDU

SendIndirectFirePDU completes and sends an Indirect Fire PDU, and assesses any damage caused to CCVs by the detonation. At the time of the call, the PDU is partially assembled in the buffer indBuf. *quantity* is the number of detonations to send on the net. *delays* is the delay, in seconds, between subsequent detonations. The function call is SendIndirectFirePDU(quantity, delays). Table 2.1-62 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
quantity	int	Standard C type.
delays	array of unsigned char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register int	Standard C type.
Calls		
Function	**Where Described**	
AssocSendDatagram	Section 2.20.1.2.1.	
PRO_SIM_IND_FIRE_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	
AssocError	Section 2.20.1.10.1.	
IndirectFireDamage	Section 2.1.2.4.6.	
Called By		
Function	**Where Described**	
PlaceIndirectFire	Section 2.1.2.4.4.	

Table 2.1-62: SendIndirectFirePDU Information.

2.1.2.4.6 IndirectFireDamage

IndirectFireDamage assesses the damage done to our CCVs by indirect fire. *pdu* is an IndirectFireVariant PDU which contains information about a series of indirect (artillery or mortar) fir detonations which occur at some location on the terrain. The routine checks the packet to see if any of the detonations are close enough to local CCVs in order to damage them. The function call is IndirectFireDamage(pdu). Table 2.1-63 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
pdu	pointer to IndirectFireVariant	/simnet/common/include/protocol/p_sim.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
det	register pointer to IndirectFireDetonation	/simnet/common/include/protocol/p_sim.h
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
i	register int	Standard C type.
detIdx	unsigned short	Standard C type.
accumDelay	unsigned short	Standard C type.
numberNearby	unsigned short	Standard C type.
ccvList	array numberCCVs of unsigned short	Standard C type.
roll	int	Standard C type.
delta	array 3 of float	Standard C type.
distance	float	Standard C type.
floatLoc	array 3 of float	Standard C type.
Calls		
Function	**Where Described**	
NearbyCCVs	Section 2.1.2.2.4.	
AttackCCV	Section 2.1.2.6.4.	
Called By		
Function	**Where Described**	
SendIndirectFirePDU	Section 2.1.2.4.5.	
ProcessSimDatagram	Section 2.1.6.2.9.	

Table 2.1-63: IndirectFireDamage Information.

2.1.2.5 placement.c

/simnet/mcc/Mother/placement.c

placement.c contains routines for the "intelligent" placement of CCVs and simulators.

2.1.2.5.1 FindAnOpenSpot

FindAnOpenSpot finds a suitable location for the placement of a vehicle. *vehicleID* is the identifier of the vehicle to be placed, *location* is the point on the terrain about which to search and *yaw* is the azimuth of the vehicle to be placed. The function returns 1 if a spot

is found, 0 if not. *location* is changed to the spot found for the vehicle and *rotation* is the orientation of the vehicle if it is placed at that spot. A suitable spot for placement is one with no other vehicles or terrain objects within the distance *vehicleRaduis* (6 meters), and with a navigable slope and soil type. An error is signalled in the case where a terrain lookup is perfomed to check for obstacles or ground height above sea level, and the terrain database code (libtdb) returns an error or undefined information. The function call is FindAnOpenSpot(vehicleID, location, yaw, rotation). Table 2.1-64 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
location	register array of float	Standard C type.
yaw	double	Standard C type.
rotation	3 by 3 matrix of double	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
objectCount	int	Standard C type.
soilType	int	Standard C type.
i	short	Standard C type.
tryCount	short	Standard C type.
direction	short	Standard C type.
step	short	Standard C type.
terrainBounds	2 by 2 matrix of float	Standard C type.
stepSize	float	Standard C type.
testLocatiion	array 2 of float	Standard C type.
tdbLocation	TDB_POINT	/simnet/common/libsrc/libtdb/tdb.h
spiral	8 by 2 matrix of char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Unsuccessful.
1	int	Successful
Errors		
Error Name	**Reason for Error**	
MCC_BAD_TERRAIN	Error from libtdb calls.	
Calls		
Function	**Where Described**	
NearbyVehicle	Section 2.1.2.5.2.	
Act_List_Loc_Exists	Section 2.1.4.1.6.	
tdb_object_count	Section 2.21.7.26.2.	
tdb_error	Section 2.21.7.17.1.	
Mines_Point_In_Minefield	Section 2.1.8.3.13.	
tdb_get_hull_to_world	Section 2.21.7.20.1.	
tdb_get_z	Section 2.21.7.16.2.	
SoilOkay	Section 2.1.3.1.1.	

Called By	
Function	**Where Described**
PlaceCCV	Section 2.1.2.6.1.
ActivateVehicle	Section 2.1.4.2.8.

Table 2.1-64: FindAnOpenSpot Information.

2.1.2.5.2 NearbyVehicle

NearbyVehicle tests for vehicles near a specified location, determined by *location* and *range*, excluding from consideration a particular vehicle (usually the one being placed), specified by *vehicleID*. The function call is NearbyVehicle(location, vehicleID, range). Table 2.1-65 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
location	register array of float	Standard C type.
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
range	float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
VA	register pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
vlist_ptr	pointer to FQueue_t	/simnet/mcc/libmcc/fqueue.h
dx	float	Standard C type.
dy	float	Standard C type.
vehLocation	array 2 of float	Standard C type.
i	unsigned short	Standard C type.
ccvList	array numberCCVs of unsigned short	Standard C type.
vehicleGuises	VehicleGuises	/simnet/common/include/protocol/basic.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	No vehicle within range.
1	int	Vehicle within range.
Calls		
Function	**Where Described**	
NearbyCCVs	Section 2.1.2.2.4.	
FQueue_Scan	Section 2.21.1.13.6.	
GetVehicleLocation	Section 2.1.1.5.4.	
IsGroundVehicle	Section 2.1.1.5.5.	

Called By	
Function	**Where Described**
FindAnOpenSpot	Section 2.1.2.5.1.

Table 2.1-65: NearbyVehicle Information.

2.1.2.5.3 NearestCombatVehicle

NearestCombatVehicle finds the ground combat vehicle nearest to a specified location. *location* is the place on the terrain to search around and *range* determines how far out to search for a combat vehicle. The function returns 1 if a vehicle is found and 0 otherwise. *location* gets the location of the combat vehicle. The function call is NearestCombatVehicle(location, range). Table 2.1-66 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
location	array of float	Standard C type.
range	float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
i	unsigned short	Standard C type.
v ehicleFound	unsigned short	Standard C type.
vehicleGuises	*VehicleGuises*	/simnet/common/include/protocol/basic.h
dx	float	Standard C type.
dy	float	Standard C type.
vehLocation	array 2 of float	Standard C type.
distance	float	Standard C type.
closestDistance	float	Standard C type.
closestLocation	array 2 of float	Standard C type.
vlist_ptr	pointer to FQueue_t	/simnet/mcc/libmcc/fqueue.h
Return Values		
Return Value	**Type**	**Meaning**
vehicleFound	int	1 if found, 0 if not.
Calls		
Function	**Where Described**	
FQueue_Scan	Section 2.21.1.13.6.	
GetVehicleLocation	Section 2.1.1.5.4.	
IsGroundVehicle	Section 2.1.1.5.5.	
IsCombatVehicle	Section 2.1.1.5.6.	
Called By		
Function	**Where Described**	
PlaceCCV	Section 2.1.2.6.1.	

Table 2.1-66: NearestCombatVehicle Information.

2.1.2.6 ccv_change.c

/simnet/mcc/Mother/ccv_change.c

ccv_change.c contains routines for making CCVs visible and invisible, and for damaging them.

2.1.2.6.1 PlaceCCV

PlaceCCV places a CCV on the terrain, making it visible. *requestID* is a code supplied by YUMM, used in returning a response to the message that requested the placement of the CCV. *msg* is the showCCV message requesting the placement of the CCV. It may not be possible to place the CCV at the location requested. There may be another vehicle that is too close, or the terrain may be too steep or impassable. Also, HEMMTs and Repair vehicles have an "affinity" for Combat vehicles. The actual location will be determined by PlaceCCV, and the location field of the mesage will be changed to this new location. An entry for the CCV is put into the vehicle table. If the CCV is on the terrain, an entry is made in the Population Density Map, and the CCV's state is set to ccvHealthy. Otherwise, its state is ccvDisappearing. This routine signals errors in the cases where preallocated memory has run out for either the vehicle status list or the vehicle ID table. This error indicates that the MCC is attempting to simulate too many vehicles. The function call is PlaceCCV(requestID, msg). Table 2.1-67 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
requestID	long	Standard C type.
msg	register pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ccv	pointer to CCVStatus	/simnet/mcc/include/veh_table.h
i	register short	Standard C type.
j	register short	Standard C type.
rotation	3 by 3 matrix of double	Standard C type.
tmp_veh	VehicleIDStatus	/simnet/mcc/include/veh_table.h
status	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_NO_VEHICLE_ID	No more vehicles IDs are available.	
MCC_VST_QUE_NOMEM	The vehicle status list is out of space.	

Calls	
Function	**Where Described**
FQueue_Insert	Section 2.21.1.13.4.
VehicleIsGunneryTarget	Section 2.1.2.6.2.
NearestCombatVehicle	Section 2.1.2.5.3.
FindAnOpenSpot	Section 2.1.2.5.1.
mil_to_rad	Macro defined in /simnet/common/include/global/sim_macros.h.
DeactivateVehicle	Section 2.1.4.2.11.
PDMInsert	Section 2.1.2.2.2.
SendRsp	Section 2.21.2.10.8.
Called By	
Function	**Where Described**
ProcessMessage	Section 2.1.1.2.1.

Table 2.1-67: PlaceCCV Information.

2.1.2.6.2 VehicleIsGunneryTarget

VehicleIsGunneryTarget returns TRUE if a vehicle has no capabilities. *capabilites* specifies whether the vehicle has the ability to recover, resupply or repair vehicles on the battlefield. If a CCV has any of these capabilities then it is NOT a gunnery target. If it has none of these capabilities, then the vehicle is an MCC gunnery target. The function call is VehicleIsGunneryTarget(capabilities). Table 2.1-68 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
capabilities	pointer to VehicleCapabilites	/simnet/common/include/protocol/basic.h
Return Values		
Return Value	**Type**	**Meaning**
FALSE	int	Vehicle is not a gunnery target.
TRUE	int	Vehicle is a gunnery target.
Called By		
Function	**Where Described**	
PlaceCCV	Section 2.1.2.6.1.	

Table 2.1-68: VehicleIsGunneryTarget Information.

2.1.2.6.3 RemoveCCV

RemoveCCV takes a CCV off of the terrain. *vehicle* refers to the vehicle to be removed. The function call is RemoveCCV(vehicle). Table 2.1-69 describes the parameter used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	unsigned short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.1.1.2.1.	

Table 2.1-69: RemoveCCV Information.

2.1.2.6.4 AttackCCV

AttackCCV notes an attack on a CCV occuring now or in the immediate future. *ccv* is the attacked CCV, *delay* is the time until the attack in ticks, *agentID* is the vehicle ID of the attacker, *eventID* is the event identifier and *cause* is the reason for the demise of the CCV. The function call is AttackCCV(ccv, delay, agentID, eventID, cause). Table 2.1-70 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
delay	unsigned short	Standard C type.
agentID	pointer toVehicleID	/simnet/common/include/protocol/basic.h
eventID	unsigned short	Standard C type.
cause	unsigned char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
time	long	Standard C type.
Called By		
Function	**Where Described**	
IndirectFireDamage	Section 2.1.2.4.6.	
ProcessSimTransaction	Section 2.1.6.2.6.	

Table 2.1-70: AttackCCV Information.

2.1.3 Positioning Vehicles

This second level CSC provides a routine to determine if a location is suitable to position a vehicle based on the soil type and pitch of the terrain. It conains one CSU, soil.c.

2.1.3.1 soil.c

/simnet/mcc/Mother/soil.c

soil.c contains a routine for checking soil versus pitch and cant.

2.1.3.1.1 SoilOkay

SoilOkay determines if the soil, given by *soilType*, is acceptable for a given vehicle at a given pitch and cant, determined by *rotation*. The function call is SoilOkay(soilType, rotation). Table 2.1-71 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
soilType	int	Standard C type.
rotation	3 by 3 matrix of double	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Soil is unacceptable.
1	int	Soil is acceptable.
Calls		
Function	**Where Described**	
PITCH_SIN	Macro defined in /simnet/mcc/Mother/soil.c.	
CANT_SIN	Macro defined in /simnet/mcc/Mother/soil.c.	
Called By		
Function	**Where Described**	
FindAnOpenSpot	Section 2.1.2.5.1.	

Table 2.1-71: SoilOkay Information.

2.1.4 Activating Combat Vehicle Simulators

This keeps a separate list of activate locations which are used to determine whether a vehicle was initialized at a particular location from this MCC, but is not yet issuing vehicle appearance packets. This is done to prevent more than one vehicle from being placed in the same location. It activates combat vehicle simulators and ascertains whether they are being activated for the first time, due to a reconstitution, or due to a towing arrival. The type of activation is used to determine the state and status of the vehicle. The following CSUs perform these functions:

actloc.c
sim.c

2.1.4.1 actloc.c

/simnet/mcc/Mother/actloc.c

actloc.c contains routines which keep track of where vehicles have been activated. This prevents one vehicle from being activated on top of another vehicle before the first has had a chance to send out appearance packets. Table 2.1-72 shows the variables used by actloc.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
Act_List	pointer to FQueue_Head_t	/simnet/mcc/libmcc/fqueue.h

Table 2.1-72: actloc.c Variable Information.

2.1.4.1.1 Act_List_Hash

Act_List_Hash is a hash function used to index the activation entries. *key* is a pointer to three unsigned integers which are used to lookup the activating location of the vehicle. *Hash_Size* is a pointer to an integer which is the size of the hash table used to lookup the entries in the activation location table. The function call is Act_List_Hash(Key, Hash_Size). Table 2.1-73 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Key	register pointer to unsigned short	Standard C type.
Hash_Size	pointer register long	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
index	register unsigned long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
index	unsigned short	Index of activation entry.

Table 2.1-73: Act_List_Hash Information.

2.1.4.1.2 Act_List_Alloc

Act_List_Alloc returns a pointer to memory to use for the activation table. *Size* is the size in bytes of the area of memory to allocate. *Hash_Size* is the size of the hash table. The function call is Act_List_Alloc(Size, Hash_Size). Table 2.1-74 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Size	unsigned int	Standard C type.
Hash_Size	unsigned long	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
((char)malloc (Size))	pointer to char	Pointer to memory area allocated.

Table 2.1-74: Act_List_Alloc Information.

2.1.4.1.3 Act_List_Init

Act_List_Init initializes the activation list. The function call is Act_List_Init(). Table 2.1-75 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dummy	pointer to int	Standard C type.
Calls		
Function	**Where Described**	
FQueue_Create	Section 2.21.1.13.3.	
Called By		
Function	**Where Described**	
Init_MCC_Processing	Section 2.1.1.2.8.	

Table 2.1-75: Act_List_Init Information.

2.1.4.1.4 Act_List_Add

Act_List_Add adds a vehicle to the list of activation locations of which to keep track. *VehicleID* is the vehicle ID of the vehicle to add to the list. It specifies uniquely the vehicle being saved. *Location* is the location of the vehicle to be stored. The function call is Act_List_Add(VehicleID, Location). Table 2.1-76 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
VehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
Location	array of float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
Act_Entry	Activate_Entry _t	/simnet/mcc/Mother/actloc.c
status	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	char	Successful.
-1	char	Unsuccessful.

Calls	
Function	**Where Described**
FQueue_Insert	Section 2.21.1.13.4.
Called By	
Function	**Where Described**
ActivateVehicle	Section 2.1.4.2.8.

Table 2.1-76: Act_List_Add Information.

2.1.4.1.5 Act_List_Delete

Act_List_Delete removes an entry from the activation list. The function call is Act_List_Delete(VehicleID). Table 2.1-77 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
VehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
status	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
-1	char	Unsuccessful.
0	char	Successful.
Calls		
Function	**Where Described**	
FQueue_Remove	Section 2.21.1.13.5.	
Called By		
Function	**Where Described**	
ActivateReplyTimedOut	Section 2.1.4.2.7.	
CheckSimulatorActivity	Section 2.1.4.2.12	

Table 2.1-77: Act_List_Delete Information.

2.1.4.1.6 Act_List_Loc_Exists

Act_List_Loc_Exists determines if a given location is near a vehicle which is already activating. The function call is Act_List_Loc_Exists(Location, Radius). Table 2.1-78 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Location	array of float	Standard C type.
Radius	unsigned int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
Entry_Ptr	pointer to FQueue_Head_t	/simnet/mcc/libmcc/fqueue.h
Activate_Entry	Activate_Entry_t	/simnet/mcc/Mother/actloc.c

Return Values		
Return Value	**Type**	**Meaning**
0	char	Location not near activating vehicle.
-1	char	Location near activating vehicle.

Calls	
Function	**Where Described**
FQueue_Scan	Section 2.21.1.13.6.
DISTANCE	Macro defined in /simnet/mcc/Mother/actloc.c.

Called By	
Function	**Where Described**
FindAnOpenSpot	Section 2.1.2.5.1.

Table 2.1-78: Act_List_Loc_Exists Information.

2.1.4.2 sim.c

/simnet/mcc/Mother/sim.c

sim.c contains routines for activating and deactivating combat simulators.

2.1.4.2.1 UpdateAppearance

UpdateAppearance is a dummy function. The function call is UpdateAppearance(vehicleID, PDU). Table 2.1-79 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
PDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
angle	float	Standard C type.
rot	float	Standard C type.
rotation	3 by 3 matrix of double	Standard C type.

Table 2.1-79: UpdateAppearance Information.

2.1.4.2.2 ActivateForExerciseStart

ActivateForExerciseStart activates a simulator at the start of an exercise. *requestID* is the identifier marking this request for activation, and is used to send a reply message back to the requestor. *msg* is the "act" structure part of MCCMessageBuffer used to lookup and set information needed for activating a simulator on the network. *sim* is a pointer to the internal simulator status maintained by the MCC for the activated simulator. The function call is ActivateForExerciseStart(requestID, msg, sim). Table 2.1-80 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
requestID	long	Standard C type.
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
sim	pointer to pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tmpsim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
i	register int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
1	int	Simulator already placed and can't be activated.
0	int	Simulator activated.
Calls		
Function	**Where Described**	
SendRsp	Section 2.21.2.10.8.	
Called By		
Function	**Where Described**	
ActivateVehicle	Section 2.1.4.2.8.	

Table 2.1-80: ActivateForExerciseStart Information.

2.1.4.2.3 ActivateForReconstitution

ActivateForReconstitution activates a vehicle that is being reconstituted. *requestID* is the identifier marking this request for activation, and is used to send a reply message back to the requestor. *msg* is the "act" structure part of MCCMessageBuffer used to lookup and set information needed for activating a simulator on the network. *sim* is a pointer to the internal simulator status maintained by the MCC for the activated simulator. The function call is ActivateForReconstitution(requestID, msg, sim). Table 2.1-81 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
requestID	long	Standard C type.
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
sim	pointer to pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
tmpsim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
1	int	Unsuccessful.
0	int	Successful.
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
SendRsp	Section 2.21.2.10.8.	
Called By		
Function	**Where Described**	
ActivateVehicle	Section 2.1.4.2.8.	

Table 2.1-81: ActivateForReconstitution Information.

2.1.4.2.4 ActivateForTowingArrival

ActivateForTowingArrival activates a vehicle when it reaches it towing destination. *requestID* is the identifier marking this request for activation, and is used to send a reply message back to the requestor. *msg* is the "act" structure part of MCCMessageBuffer used to lookup and set information needed for activating a simulator on the network. It is a dummy function. The function call is ActivateForTowingArrival(requestID, msg). Table 2.1-82 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
requestID	long	Standard C type.
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	Always returned.

Called By	
Function	**Where Described**
ActivateVehicle	Section 2.1.4.2.8.

Table 2.1-82: ActivateForTowingArrival Information.

2.1.4.2.5 Build_ActivatePDU

Build_ActivatePDU creates an Activate PDU for a simulator. *PDU* is the ActivateRequestVariant simulation PDU. *msg* is the MCCMessageBuffer containing the "act" structure used by this routine for information about this activation attempt. *simAddress* is the SIMNET simulator address of the vehicle being activated. *bumper* is the bumper number (a license plate type of number) used by this routine to fill in part of the "text" part of the "marking" field in the activation variant. The function call is Build_ActivatePDU(PDU, msg, simAddress, location). Table 2.1-83 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to Simulation PDU	/simnet/common/include/protocol/p_sim.h
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
simAddress	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
location	array of float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
aPDU	register pointer to ActivateRequestVariant	/simnet/common/include/protocol/p_sim.h
vaPDU	pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
unitID	UnitIdentifier	/simnet/common/include/protocol/basic.h
bumper	pointer to unsigned int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
1	int	Unsuccessful.
0	int	Successful.
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
mil_to_fixedpt	Section 2.21.1.5.1.	

Called By	
Function	**Where Described**
ActivateVehicle	Section 2.1.4.2.8.

Table 2.1-83: Build_ActivatePDU Information.

2.1.4.2.6 ActivateReplyReceived

ActivateReplyReceived is called when a response is received to a Activate request transaction. *PDU* is the activate response variant received when the simulator being activated responds to the MCC activate request. *PDU_Len* is the size of the received PDU in bytes. *Respondent* is the SimulatorAddress of the simulator responding to the activate request. *Act_Info* contains information about the simulator's entry in the MCC local simulator table. The function call is ActivateReplyReceived(PDU, PDU_Len, Respondent, Act_Info). Table 2.1-84 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	long int	Standard C type.
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Act_Info	pointer to VMisc_Data_t	/simnet/mc/Mother/sim.c
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
aPDU	register pointer to ActivateResponseVariant	/simnet/common/include/protocol/p_sim.h
tmp_veh	VehicleIDStatus	/simnet/mcc/include/veh_table.h
status	int	Standard C type.
Calls		
Function	**Where Described**	
FQueue_Insert	Section 2.21.1.13.4.	
FQueue_Retrieve	Section 2.21.1.13.7.	

Table 2.1-84: ActivateReplyReceived Information.

2.1.4.2.7 ActivateReplyTimedOut

ActivateReplyTimedOut is called when a simulator which the MCC tried to activate did not send a reply. *PDU* is the activate request variant sent to the simulator. *PDU_Len* is the size of the PDU in bytes. *Respondent* is the SimulatorAddress of the simulator being requested to activate. *Act_Info* contains information about the simulator's entry in the MCC local simulator table. The function call is ActivateReplyTimedOut(PDU, PDU_Len, Respondent, Act_Info). Table 2.1-85 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	long int	Standard C type.
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Act_Info	pointer to VMisc_Data_t	/simnet/mcc/Mother/sim.c

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
aPDU	register pointer to ActivateRequestVariant	/simnet/common/include/protocol/p_sim.h
tmp_id	VehicleID	/simnet/common/include/protocol/basic.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h

Calls	
Function	**Where Described**
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.
VehicleToIDTrailer	Section 2.21.1.11.4.
Act_List_Delete	Section 2.1.4.1.5.

Table 2.1-85: ActivateReplyTimedOut Information.

2.1.4.2.8 ActivateVehicle

ActivateVehicle activates a vehicle simulator. *requestID* is an identifier to use when returning a response to the YUMM request asking for activation and *msg* is the message sent to Mother asking for the activation. If the vehicle is being activated because it is arriving at a towing destination, only its location and orientation (not status) need to be supplied by the requesting process.

Errors are signalled for the following conditions:

-- No clear place on the terrain can be found to place the vehicle
-- The reason for activation is not among those known to the MCC
-- The ActivateForExerciseStart, ActivateForReconstitution, or ActivateForTowingArrival routines return a non-zero status
-- Some part of the activation information is invalid, as returned by Build_ActivatePDU
-- There is no room in the activation list to add the location of the vehicle
-- The network cannot be accessed successfully

The function call is ActivateVehicle(requestID, msg). Table 2.1-86 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
requestID	long	Standard C type.
msg	register pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tmpmsg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
vehicleID	VehicleID	/simnet/common/include/protocol/basic.h
PDU	SimulationPDU	/simnet/common/include/protocol/p_sim.h
simAddress	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
Act_Info	pointer to VMisc_Data_t	/simnet/mcc/Mother/sim.c
location	array 3 of float	Standard C type.
theta	double	Standard C type.
rotation	3 by 3 matrix of double	Standard C type.
status	int	Standard C type.
unitID	pointer to UnitIdentifier	/simnet/common/include/protocol/basic.h

Errors	
Error Name	**Reason for Error**
MCC_ACTIVATE_FAILED	The simulator replied to the activate request but did not start.
MCC_ACTIVATE_BAD	Corrupt activate sent to simulator.
MCC_ACTIVATE_ASSFAIL	Send failed activating.

Calls	
Function	***Where Described***
FindAnOpenSpot	Section 2.1.2.5.1.
mil_to_rad	Macro defined in /simnet/common/include/global/sim_macros.h.
SendRsp	Section 2.21.2.10.8.
ActivateForExerciseStart	Section 2.1.4.2.2.
ActivateForReconstitution	Section 2.1.4.2.3.
ActivateForTowingArrival	Section 2.1.4.2.4.
VehicleIDToTrailer	Section 2.21.1.11.4.
TotalVehicleFuel	Section 2.21.1.28.1.
Build_ActivatePDU	Section 2.1.4.2.5.
FQueue_Retrieve	Section 2.21.1.13.7.
AssocSendTransact	Section 2.20.1.4.1.
PRO_SIM_ACTIVATE_REQ_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h
AssocError	Section 2.20.1.10.1.
Act_List_Add	Section 2.1.4.1.4.

Called By	
Function	**Where Described**
ProcessMessage	Section 2.1.1.2.1.

Table 2.1-86: ActivateVehicle Information.

2.1.4.2.9 DeactivateReplyReceived

DeactivateReplyReceived is called when a response is received to a deactivate request. *PDU* is the deactivate response variant received when the simulator being deactivated responds to the MCC deactivate request. *PDU_Len* is the size of the received PDU in bytes. *Respondent* is the SimulatorAddress of the simulator responding to the deactivate request. *Deact_Info* contains information about the simulator's entry in the MCC local simulator table. An error is signalled if the MCC received a response from a simulator that was not currently being deactivated. The function call is DeactivateReplyReceived(PDU, PDU_Len, Respondent, Deact_Info). Table 2.1-87 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to Simulation PDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	unsigned int	Standard C type.
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Deact_Info	pointer to VMisc_Data_t	/simnet/mcc/Mother/sim.c
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dPDU	pointer to DeactivateResponseVariant	/simnet/common/include/protocol/p_sim.h
Errors		
Error Name	***Reason for Error***	
MCC_DEACTIVATE_BASRSP	Deactivate failed.	

Table 2.1-87: DeactivateReplyReceived Information.

2.1.4.2.10 DeactivateReplyTimedOut

DeactivateReplyTimedOut is called when no response is received to a deactivate request. *PDU* is the deactivate request sent to the simulator being by the MCC. *PDU_Len* is the size of the PDU in bytes. *Respondent* is the SimulatorAddress of the simulator being deactivated. *Deact_Info* contains information about the simulator's entry in the MCC local simulator table. The function call is DeactivateReplyTimedOut(PDU, PDU_Len, Respondent, Deact_Info). Table 2.1-88 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to Simulation PDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	unsigned int	Standard C type.
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Deact_Info	pointer to VMisc_Data_t	/simnet/mcc/Mother/sim.c

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dPDU	pointer to DeactivateRequestVariant	/simnet/common/include/protocol/p_sim.h
Errors		
Error Name	**Reason for Error**	
MCC_DEACTIVATE_FAILED	Simulator could not be deactivated.	
Calls		
Function	**Where Described**	
VehicleToIDTrailer	Section 2.21.1.11.4.	

Table 2.1-88: DeactivateReplyTimedOut Information.

2.1.4.2.11 DeactivateVehicle

DeactivateVehicle deactivates a vehicle simulator, *vehicleID*, by broadcasting the appropriate number of Deactivate PDUs. The function call is DeactivateVehicle(vehicleID). Table 2.1-89 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
PDU	SimulationPDU	/simnet/common/include/protocol/p_sim.h
dPDU	pointer to DeactivateRequestVariant	/simnet/common/include/protocol/p_sim.h
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
status	int	Standard C type.
Deact_Info	pointer to VMisc_Data_t	/simnet/mcc/Mother/sim.c
Errors		
Error Name	**Reason for Error**	
MCC_DEACTIVATE_ASSFAIL	Send failed deactivating.	
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
AssocSendTransact	Section 2.20.1.4.1.	
PRO_SIM_DEACTIVATE_REQ_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	
AssocSendDatagram	Section 2.20.1.2.1.	
AssocError	Section 2.20.1.10.1.	

Called By	
Function	**Where Described**
ProcessMessage	Section 2.1.1.2.1.
BroadcastCCVBatch	Section 2.1.2.3.2.
PlaceCCV	Section 2.1.2.6.1.
DeactivateAllVehicles	Section 2.1.5.1.2

Table 2.1-89: DeactivateVehicle Information.

2.1.4.2.12 CheckSimulatorActivity

CheckSimulatorActivity monitors the receipt of Vehicle Appearance PDUs. An error is signalled if a simulator which the MCC activated suddenly stops broadcasting packets on the network, and a message is sent to the SCC indicating that the simulator may have some problem. The function call is CheckSimulatorActivity(). Table 2.1-90 describes the internal variables used, errors returned and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
vlist_ptr	pointer to FQueue_t	/simnet/mcc/libmcc/fqueue.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
vaPDU	pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
timedOut	int	Standard C type.
OldPacket	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
checkSimulatorPeriod	int	Delay time until fuction can be called again. checkSimulatorPeriod is a constant defined in /simnet/mcc/include/MCC_timers.h
Errors		
Error Name	**Reason for Error**	
MCC_SIM_LOST	Simulator stopped unexpectedly.	
Calls		
Function	**Where Described**	
FQueue_Scan	Section 2.21.1.13.6.	
FQueue_Retrieve	Section 2.21.1.13.7.	
Act_List_Delete	Section 2.1.4.1.5.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h	
VehicleIDToTrailer	Section 2.21.1.11.4.	
FQueue_Remove	Section 2.21.1.13.5.	

Table 2.1-90: CheckSimulatorActivity Information.

2.1.5 Recording Statistics

This writes out two files at the end of the exercise: ExerciseLog and VehicleLog. These keep track of information during the exercise. The start and end times of the exercise are noted, and certain vehicle information, e.g. ammo used, distance travelled, etc. are traced. This functionality is provided by one CSU, shutdown.c.

2.1.5.1 shutdown.c
/simnet/mcc/Mother/shutdown.c

shutdown.c contains several routines which are called during exercise shutdown, MCC system exit and host powerdown.

2.1.5.1.1 ResetMCC

ResetMCC completely resets the MCC and ends the current exercise if one exists. *fast* is normally 0 when in a normal reset or shutdown of the MCC. Generally the MCC will send out deactivate packets to vehicles which it has activated when shut down. When a "fast" shutdown is specified, the MCC skips the phase of deactivating simulators and the simulators remain in their previous state. The function call is ResetMCC(fast). Table 2.1-91 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
fast	int	Standard C type.
Calls		
Function	**Where Described**	
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
KillServiceProcesses	Section 2.1.5.1.3.	
LogExerciseEnd	Section 2.1.5.1.4.	
DeactivateAllVehicles	Section 2.1.5.1.2.	
tdb_terminate	Section 2.21.7.27.2.	
YUMMCleanUp	Section 2.21.2.10.5.	
RemoveLocks	Section 2.21.2.6.5.	
RemoveSharedMem	Section 2.21.2.7.2.	
CloseNetworkInterface	Section 2.21.1.22.4.	
Called By		
Function	**Where Described**	
main	Section 2.1.1.2.9.	

Table 2.1-91: ResetMCC Information.

2.1.5.1.2 DeactivateAllVehicles

DeactivateAllVehicles deactivates each vehicle currently active. The function call is DeactivateAllVehicles(). Table 2.1-92 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
j	int	Standard C type.
sim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
DeactivateVehicle	Section 2.1.4.2.11.	
Called By		
Function	**Where Described**	
ResetMCC	Section 2.1.5.1.1.	

Table 2.1-92: DeactivateAllVehicles Information.

2.1.5.1.3 KillServiceProcesses

KillServiceProcesses terminates each of the service processes spawned by the Mother process. The function call is KillServiceProcesses(). Table 2.1-93 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
errno	extern int	Standard C type.
Calls		
Function	**Where Described**	
SignalChild	Section 2.21.1.26.4.	
Called By		
Function	**Where Described**	
ResetMCC	Section 2.1.5.1.1.	

Table 2.1-93: KillServiceProcesses Information.

2.1.5.1.4 LogExerciseEnd

LogExerciseEnd logs the statistics to log files following an exercise. The function call is LogExerciseEnd(). Table 2.1-94 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
cp	register pointer to char	Standard C type.
i	short	Standard C type.
j	short	Standard C type.
totalDistance	float	Standard C type.
totalFuel	float	Standard C type.
totalAmmo	array numberAmmoCounters of long	Standard C type.
dateString	array 20 of char	Standard C type.
startTime	array 20 of char	Standard C type.
endTime	array 20 of char	Standard C type.
exerciseKey	array 20 of char	Standard C type.

Calls	
Function	**Where Described**
TimeString	Section 2.1.5.1.5.
FQueue_Retrieve	Section 2.21.1.13.7.

Called By	
Function	**Where Described**
ResetMCC	Section 2.1.5.1.1.

Table 2.1-94: LogExerciseEnd Information.

2.1.5.1.5 TimeString

TimeString creates a string representation of a date and time. *Time* is the current time in seconds since Jan 1, 1970. *dateString* creates the date of *Time* as MM.DD.YY and returns it. *timeString* creates the time of *Time* as HH.MM.SS and returns it. The function call is TimeString(time, dateString, timeString). Table 2.1-95 describes the parameters used, and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
time	long	Standard C type.
dateString	pointer to char	Standard C type.
timeString	pointer to char	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tm	pointer to struct tm	Standard C structure.

Called By	
Function	**Where Described**
LogExerciseEnd	Section 2.1.5.1.4.

Table 2.1-95: TimeString Information.

2.1.6 Listening to the SIMNET Local Area Network

This is the CSC which reads packets in from the SIMNET local area network and either uses them or distributes them to the appropriate console process. It saves vehicle status packets and vehicle appearance packets to update its knowledge of the state of the world. It also listens to impact packets and determines if any of its CCVs have been hit. It listens to resupply packets to pass onto Maint or Admin as well as other packets described in the SIMNET MCC Report. It is composed of the following CSUs:

maint.c
protocol.c

2.1.6.1 maint.c

/simnet/mcc/Mother/maint.c

maint.c contains routines which handle repair/service maintenance functions. Table 2.1-96 shows the variables used by maint.c.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
networkInterface	extern int	Standard C type.

Table 2.1-96: maint.c Variable Information.

2.1.6.1.1 RepairReplyReceived

RepairReplyReceived is a dummy function. The function call is RepairReplyReceived(PDU, PDU_Len, Respondent, Param). Table 2.1-97 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to Simulation PDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	long int	Standard C type.
Respodent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Param	pointer to long	Standard C type.

Table 2.1-97: RepairReplyReceived Information.

2.1.6.1.2 RepairReplyTimedOut

RepairReplyTimedOut is a dummy function. The function call is RepairReplyTimedOut(PDU, PDU_Len, Respondent, Param). Table 2.1-98 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to Simulation PDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	long int	Standard C type.
Respodent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Param	pointer to long	Standard C type.

Table 2.1-98: RepairReplyTimedOut Information.

2.1.6.1.3 Repair_Request

Repair_Request is called when the operator of the Maint Macintosh console has directed a repair to take place. The Main console sends a sendRepairReq message to the Mother process. The request contains (within an MCCMessageBuffer) the information about the repair process. The message is dispatched on the network as a transaction. RepairReplyTimedOut() and RepairReplyReceived() are essentially placeholder routines which are invoked when the repair response or timeout is received; since the response contains no information, no processing is performed. *requestID* is the unique identifier which identifies this transaction. *msg* points to the "repair" structure in the MCCMessageBuffer structure. The repair request transaction contains the IDs of the vehicle being fixed, the vehicle doing the fixing, and the type of repair being performed. An error is signalled for the case when the network could not be accessed for some reason and a textural error message is printed on the MCC console via the AssocError() routine in libassoc. The function call is Repair_Request(requestID, msg). Table 2.1-99 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
requestID	long	Standard C type.
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
PDU	SimulationPDU	/simnet/common/include/protocol/p_sim.h
repairPDU	pointer to RepairRequestVariant	/simnet/common/include/protocol/p_sim.h
status	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_REPAIR_ASSFAIL	the network could not be accessed	
Calls		
Function	**Where Described**	
AssocSendTransact	Section 2.20.1.4.1.	
PRO_SIM_REPAIR_REQUEST_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	
AssocError	Section 2.20.1.10.1.	

Called By	
Function	**Where Described**
ProcessMessage	Section 2.1.1.2.1.

Table 2.1-99: Repair_Request Information.

2.1.6.2 protocol.c

/simnet/mcc/Mother/protocol.c

protocol.c contains routines for processing PDUs received from the network.

2.1.6.2.1 SaveVehicleStatus

SaveVehicleStatus is used to save the status of a vehicle on the network. The MCC uses the status information (fuel remaining, weapons available, fixed/broken state of various subsystems, etc) when displaying information about a simulator or activating the simulator on the terrain. The information is saved in the vehicle status list, *vStatusList*. *PDU* is a DataCollection PDU pointer containing the VehicleStatusVariant with all of the known status information about a vehicle. An error is signalled if there is not enough room in the vehicle status list to add another status entry. The function call is SaveVehicleStatus(PDU). Table 2.1-100 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
tmp_veh	VehicleIDStatus	/simnet/mcc/include/veh_table.h
fuel	float	Standard C type.
status	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_VST_QUE_NOMEM	Vehicle status list out of space.	
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
FQueue_Insert	Section 2.21.1.13.4.	
Lock	Section 2.21.2.6.3.	
Unlock	Section 2.21.2.6.4.	
TotalVehicleFuel	Section 2.21.1.28.1.	

Called By	
Function	**Where Described**
ProcessDataColDatagram	Section 2.1.6.2.2.

Table 2.1-100: SaveVehicleStatus Information.

2.1.6.2.2 ProcessDatColDatagram

ProcessDatColDatagram processes a datagram PDU pertaining to the Data Collection Protocol. *PDU* is the Data Collection Protocol Data Unit received from the network for processing by this routine. *PDU_Len* is the size of the PDU in bytes. *Group* is the multicast group ID for which this packet was received. In Simnet, the multicast group ID is made up of the protocol number and the exercise ID. *Protocol* is the dataCollectionProtocolNumber. *Originator* is the Simulation Address of the simulator which initiated the transaction. *Trans_ID* is an internal identifier used by the Association library to associate the PDU with a response which may be sent to a libassoc transaction. *Respondent* is the Simulation Address of the simulator which is supposed to respond if this is a transaction. An error is signalled if the Association library is unable to access the network. The function call is ProcessDatColDatagram(PDU, PDU_Len, Group, Protocol, Originator, Trans_ID, Respondent). Table 2.1-101 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Originator	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Calls		
Function	**Where Described**	
SaveVehicleStatus	Section 2.1.6.2.1.	
ProcessStatusQueryDatagram	Section 2.1.7.1.10.	

Table 2.1-101: ProcessDatColDatagram Information.

2.1.6.2.3 ProcessDataColTransaction

ProcessDataColTransaction processes a PDU pertaining to the Data Collection Protocol. *PDU* is the Data Collection Protocol Data Unit received from the network for processing by this routine. *PDU_Len* is the size of the PDU in bytes. *Group* is the multicast group ID for which this packet was received. In Simnet, the multicast group ID is made up of the protocol number and the exercise ID. *Protocol* is the dataCollectionProtocolNumber. *Originator* is the Simulation Address of the simulator which initiated the transaction. *Trans_ID* is an internal identifier used by the Association library to associate the PDU with a response which may be sent to a libassoc transaction. *Respondent* is the Simulation Address of the simulator which is supposed to respond to the transaction. An error is signalled if the Association library is unable to access the network. The function call is ProcessDataColTransaction(PDU, PDU_Len, Group, Protocol, Originator, Trans_ID, Respondent). Table 2.1-102 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Originator	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Calls		
Function	**Where Described**	
ProcessStatusQueryTransaction	Section 2.1.7.1.11.	

Table 2.1-102: ProcessDataColTransaction Information.

2.1.6.2.4 ProcessMgmtDatagram

ProcessMgmtDatagram is a dummy function. The function call is ProcessMgmtDatagram(PDU, PDU_Len, Group, Protocol, Originator, Trans_ID, Respondent). Table 2.1-103 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Originator	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h

Table 2.1-103: ProcessMgmtDatagram Information.

2.1.6.2.5 ProcessMgmtTransaction

ProcessMgmtTransaction is a dummy function. The function call is ProcessMgmtTransaction(PDU, PDU_Len, Group, Protocol, Originator, Trans_ID, Respondent). Table 2.1-104 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Originator	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h

Table 2.1-104: ProcessMgmtTransaction Information.

2.1.6.2.6 ProcessSimTransaction

ProcessSimTransaction processes a simulation protocol transaction. *PDU* is the Simulation Protocol Data Unit received from the network for processing by this routine. *PDU_Len* is the size of the PDU in bytes. *Group* is the multicast group ID for which this packet was received. In Simnet, the multicast group ID is made up of the protocol number and the exercise ID. *Protocol* is the simulationProtocolNumber. *Originator* is the Simulation Address of the simulator which initiated the transaction. *Trans_ID* is an internal identifier used by the Association library to associate the PDU with a response which may be sent to a libassoc transaction. *Respondent* is the Simulation Address of the simulator which is supposed to respond to the transaction. An error is signalled if the Association library is

unable to access the network. The function call is ProcessSimTransaction(PDU, PDU_Len, Group, Protocol, Originator, Trans_ID, Respondent). Table 2.1-105 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Originator	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
ccv	pointer to CCVStatus	/simnet/mcc/include/veh_table.h
status	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_SENDRSP_ASSFAIL	AssocSendResponse failed.	
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
AssocSendResponse	Section 2.20.1.4.2.	
AssocError	Section 2.20.1.10.1.	
AttackCCV	Section 2.1.2.6.4.	

Table 2.1-105: ProcessSimTransaction Information.

2.1.6.2.7 Remap_Ammo

Remap_Ammo returns a manifest type ammo from a SIMNET V6.0 ammo object, *ammo_type*. The function call is Remap_Ammo(ammo_type). Table 2.1-106 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
ammo_type	ObjectType	/simnet/common/include/protocol/basic.h

Return Values		
Return Value	**Type**	**Meaning**
ammoHEI25	int	Type of ammo.
ammoHEAT105	int	Type of ammo.
ammoAPDS25	int	Type of ammo.
ammoAPDS105	int	Type of ammo.
ammoHE107	int	Type of ammo.
ammoHE155	int	Type of ammo.
ammoMissileTOW	int	Type of ammo.
ammoMissileFAAD	int	Type of ammo.
ammoHellfire	int	Type of ammo.
ammoMaverick	int	Type of ammo.
ammoDRAGON	int	Type of ammo.
0	int	Munition type not recognized.

Table 2.1-106: Remap_Ammo Information.

2.1.6.2.8 Remap_Fuel

Remap_Fuel is a dummy function which always returns 0. The function call is Remap_Fuel(fuel_type). Table 2.1-107 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
fuel_type	ObjectType	/simnet/common/include/protocol/basic.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	Always returned.

Table 2.1-107: Remap_Fuel Information.

2.1.6.2.9 ProcessSimDatagram

ProcessSimDatagram processes a PDU pertaining to the Simulation Protocol. The function call is ProcessSimDatagram(PDU, PDU_Len, Group, Protocol, Originator, Trans_ID, Respondent). Table 2.1-108 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h
PDU_Len	long	Standard C type.
Group	MulticastGroupID	/simnet/common/include/protocol/p_assoc.h
Protocol	AssociationUserProtocol	/simnet/common/include/protocol/p_assoc.h
Originator	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h
Respondent	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
ccv	register pointer to CCVStatus	/simnet/mcc/include/veh_table.h
supplierID	VehicleID	/simnet/mcc/include/veh_table.h
supplier	unsigned short	Standard C type.
sup	pointer to MunitionQuantity	/simnet/common/include/protocol/basic.h
i	short	Standard C type.
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Errors		
Error Name	**Reason for Error**	
MCC_BAD_SUPPLY	Vehicle list unknown supply class.	
Calls		
Function	**Where Described**	
FQueue_Insert	Section 2.21.1.13.4.	
FQueue_Retrieve	Section 2.21.1.13.7.	
CountRoundsFired	Section 2.1.6.2.10.	
IndirectFireDamage	SeeSection 2.1.2.4.6.	
SendMCCMsg2	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Mines_Process_Breach_Datagram	Section 2.1.8.3.10.	

Table 2.1-108: ProcessSimDatagram Information.

2.1.6.2.10 CountRoundsFired

CountRoundsFired records the number of rounds fired by a simulator for which this MCC is responsible. *vehicleID* is a pointer to the vehicle ID of a vehicle which has just fired off a weapon. *burst* specifies the type and quantity of munitions fired. The function call is CountRoundsFired(vehicleID, burst). Table 2.1-109 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
burst	pointer to BurstDescriptor	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
Calls		
Function	**Where Described**	
VehicleIDToSim	Section 2.1.1.5.3.	
Called By		
Function	**Where Described**	
ProcessSimDatagram	Section 2.1.6.2.9.	

Table 2.1-109: CountRoundsFired Information.

2.1.7 Monitoring of the MCC System

This CSC monitors an event list, making sure that events are processed at the proper times. Examples of events include checking to see if simulators have dropped out of the network, checking to see if artillery is due to drop, or sending out exercise status or equipment status PDUs, etc. It also is notified if the exercise has ended and subsequently deactivates all of its vehicles and kills all of its child processes. This functionality is realized by the following CSUs:

status.c
vtimeout.c

2.1.7.1 status.c

/simnet/mcc/Mother/status.c

status.c contains routines for reporting MCC equipment and exercise status.

2.1.7.1.1 ReportMCCSimulationStatus

ReportMCCSimulationStatus sends an MCC Status PDU. An error is signalled if the Association library cannot access the network to send the MCC Status PDU. The function call is ReportMCCSimulationStatus(). Table 2.1-110 describes the internal variables used, errors returned and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
buf	array maxNetworkPDUSize of char	Standard C type.
dPDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
size	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
protocolSimulationStatusTime 1000	int	
Errors		
Error Name	**Reason for Error**	
MCC_ASSOC_SNDDGRAM	libassoc cannot access the network	
Calls		
Function	**Where Described**	
Build_Simulation_Status	Section 2.1.7.1.7.	
AssocSendDatagram	Section 2.20.1.2.1.	
AssocError	Section 2.20.1.10.1.	

Table 2.1-110: ReportMCCSimulationStatus Information.

2.1.7.1.2 same_unit

same_unit checks to see if two organizational units are the same. The function call is same_unit(other, me). Table 2.1-111 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
other	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
me	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
op	register pointer to UnitIdentifier	/simnet/common/include/protocol/basic.h
mp	register pointer to UnitIdentifier	/simnet/common/include/protocol/basic.h
i	register int	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
FALSE	int	The two units are not the same.
TRUE	int	The two units are the same.
Called By		
Function	**Where Described**	
IgnoreStatusQuery	Section 2.1.7.1.5.	

Table 2.1-111: same_unit Information.

2.1.7.1.3 included_unit

included_unit determines whether *me* is included by *other*. The function call is included_unit(other, me). Table 2.1-112 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
other	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
me	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
op	register pointer to UnitIdentifier	/simnet/common/include/protocol/basic.h
mp	register pointer to UnitIdentifier	/simnet/common/include/protocol/basic.h
i	register int	Standard C type.
last_valid_value	register int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
FALSE	int	Unit is not included in other.
TRUE	int	Unit is included in other.
Called By		
Function	**Where Described**	
IgnoreStatusQuery	SeeSection 2.1.7.1.5.	

Table 2.1-112: included_unit Information.

2.1.7.1.4 including_unit

including_unit determines if *other* is included by *me*. The function call is including_unit(other, me). Table 2.1-113 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
other	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
me	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
op	register pointer to UnitIdentifier	/simnet/common/include/protocol/basic.h
mp	register pointer to UnitIdentifier	/simnet/common/include/protocol/basic.h
i	register int	Standard C type.
last_valid_value	register int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
FALSE	int	Other is not included in me.
TRUE	int	Other is included in me.
Called By		
Function	**Where Described**	
IgnoreStatusQuery	Section 2.1.7.1.5.	

Table 2.1-113: including_unit Information.

2.1.7.1.5 IgnoreStatusQuery

IgnoreStatusQuery is invoked when the MCC has received a StatusQuery PDU from the network. The StatusQuery may specify the kind of simulations which should respond to the query. This routine determines whether or not the MCC meets the criteria specified in the StatusQuery; the routine returns 1 if the MCC can safely ignore the query and 0 if the MCC must respond to the query. *StatusQuery* is the StatusQueryVariant which contains information about what kind of status is requested by a remote simulator and what type of simulators should respond to the query. *exercise* is the exercise in which the MCC is currently being run. If the StatusQuery specifies *exercise* and the MCC's exercise is different, the StatusQuery can be ignored. The function call is IgnoreStatusQuery(sPDU, exercise). Table 2.1-114 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sPDU	pointer to StatusQueryVariant	/simnet/common/include/protocol/p_data.h
exercise	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vehIrrelevant	VehicleID	/simnet/common/include/protocol/basic.h

Return Values		
Return Value	**Type**	**Meaning**
1	int	the MCC can safely ignore the query
-1	int	an error has occurred
0	int	the MCC must respond to the query
Calls		
Function	**Where Described**	
same_unit	Section 2.1.7.1.2.	
included_unit	Section 2.1.7.1.3.	
including_unit	Section 2.1.7.1.4.	
Called By		
Function	**Where Described**	
Do_Generic_StatusQuery	Section 2.1.7.1.9.	

Table 2.1-114: IgnoreStatusQuery Information.

2.1.7.1.6 Build_Exercise_Status

Build_Exercise_Status composes the information in the Exercise Status PDU and returns the size of the PDU. *dPDU* is the DataCollectionPDU which will be sent on the network describing the real-world time, simulated time, terrain, and battle scheme being used by the MCC in its current exercise. The function call is Build_Exercise_Status(dPDU). Table 2.1-115 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dPDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
exerPDU	pointer to ExerciseStatusVariant	/simnet/common/include/protocol/p_data.h
Return Values		
Return Value	**Type**	**Meaning**
PRO_DATA_EXERCISE_STATUS_SIZE(dPDU)	int	the size of the PDU
Calls		
Function	**Where Described**	
PRO_DATA_EXERCISE_STATUS_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	

Called By	
Function	**Where Described**
Do_Generic_StatusQuery	Section 2.1.7.1.9.

Table 2.1-115: Build_Exercise_Status Information.

2.1.7.1.7 Build_Simulation_Status

Build_Simulation_Status composes the information in the SimulationStatusVariant and returns the size of the PDU. *dPDU* is a pointer to a SimulationStatusVariant and contains the information from the ExerciseStatusVariant used in the Build_Exercise_Status() routine, as well as unit organizational information and optional simulated elements specific to the MCC system. The function call is Build_Simulation_Status(dPDU). Table 2.1-116 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dPDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sPDU	pointer to SimulationStatusVariant	/simnet/common/include/protocol/p_data.h
unit	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
simnetMCC	pointer to SIMNET_MCC_Status	/simnet/common/include/protocol/p_data.h
Return Values		
Return Value	**Type**	**Meaning**
PRO_DATA_SIMULATION_STATUS_SIZE(dPDU, sPDU->numberUnits) + sizeof(SIMNET_MCC_Status)	int	the size of the PDU
Calls		
Function	**Where Described**	
PRO_DATA_SIMULATION_STATUS_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	
Called By		
Function	**Where Described**	
ReportMCCSimulationStatus	Section 2.1.7.1.1.	
Do_Generic_StatusQuery	Section 2.1.7.1.9.	

Table 2.1-116: Build_Simulation_Status Information.

2.1.7.1.8 Build_Vehicle_Status

Build_Vehicle_Status fills in the VehicleStatusVariant with information about the specified vehicle and returns the size of the constructed PDU. *dPDU* is a DataCollectionPDU containing a target VehicleStatusVariant. *vehicleID* specifies the MCC vehicle on which

information has been requested. If status information cannot be found on the target vehicle, an error is flagged. The function call is Build_Vehicle_Status(dPDU, vehicleID). Table 2.1-117 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dPDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vsPDU	VehicleStatusVariant	/simnet/common/include/protocol/p_data.h
vehicleStatus	pointer to VehicleStatus	/simnet/common/include/protocol/status.h
unit	pointer to OrganizationalUnit	/simnet/common/include/protocol/basic.h
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
Return Values		
Return Value	**Type**	**Meaning**
PRO_DATA_VEHICLE_STATUS_SIZE(dPDU)	int	the size of the constructed PDU
Errors		
Error Name	**Reason for Error**	
MCC_NO_CCV_STATUS	No CCV status for vehicle.	
Calls		
Function	**Where Described**	
PRO_DATA_VEHICLE_STATUS_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	
FQueue_Retrieve	Section 2.21.1.13.7.	
Called By		
Function	**Where Described**	
Do_Generic_StatusQuery	Section 2.1.7.1.9.	

Table 2.1-117: Build_Vehicle_Status Information.

2.1.7.1.9 Do_Generic_StatusQuery

Do_Generic_StatusQuery determines if the Status Query should be done. *sPDU* is a StatusQueryVariant which describes a kind of query which has been requested by a remote simulator. *dPDU* is a DataCollectionPDU, part of which may or may not be filled in by a specific status-filling routine. *dSize* is a pointer to the size of the DataCollection PDU, which is set to the size of the DataCollection PDU if it is filled in. *exercise* is the number of

the exercize in which the MCC is running. The function call is Do_Generic_StatusQuery(sPDU, dPDU, dSize, exercise). Table 2.1-118 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sPDU	pointer to StatusQueryVariant	/simnet/common/include/protocol/p_data.h
dPDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
dSize	pointer to int	Standard C type.
exercise	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Query ignored.
1	int	Query done.
Calls		
Function	**Where Described**	
IgnoreStatusQuery	Section 2.1.7.1.5.	
Build_Exercise_Status	Section 2.1.7.1.6.	
Build_Simulation_Status	Section 2.1.7.1.7.	
Build_Vehicle_Status	Section 2.1.7.1.8.	
Called By		
Function	***Where Described***	
ProcessStatusQueryDatagram	Section 2.1.7.1.10.	
ProcessStatusQueryTransaction	Section 2.1.7.1.11.	

Table 2.1-118: Do_Generic_StatusQuery Information.

2.1.7.1.10 ProcessStatusQueryDatagram

ProcessStatusQueryDatagram calls Do_GenericStatusQuery() to determine whether a response is warranted, and to build the appropriate Status PDU. If required, the DataCollectionPDU is sent out on the network. *PDU* is a DataCollectionPDU containing a StatusQueryVariant which may specify a query to which this MCC must respond. The function call is ProcessStatusQueryDatagram(PDU). Table 2.1-119 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sPDU	pointer to StatusQueryVariant	/simnet/common/include/protocol/p_data.h
dPDU	DataCollectionPDU	/simnet/common/include/protocol/p_data.h
dSize	int	Standard C type.
status	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_ASSOC_SNDDGRAM	Cannot access the network	
Calls		
Function	**Where Described**	
Do_Generic_StatusQuery	Section 2.1.7.1.9.	
AssocSendDatagram	Section 2.20.1.2.1.	
AssocError	Section 2.20.1.10.1.	
Called By		
Function	**Where Described**	
ProcessDatColDatagram	Section 2.1.6.2.2.	

Table 2.1-119: ProcessStatusQueryDatagram Information.

2.1.7.1.11 ProcessStatusQueryTransaction

ProcessStatusQueryTransaction is equivalent to ProcessStatusQueryDatagram except that it sends the StatusQuery generated by Do_Generic_StatusQuery back to the requesting simulator as a transaction. The additional parameters are *Originator*, which specifies the Simulation Address of the requesting simulator, and *Trans_ID*, which identifies this particular transaction. An error is signalled in the case that libassoc cannot access the network. The function call is ProcessStatusQueryTransaction(PDU, Originator, Trans_ID). Table 2.1-120 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
PDU	pointer to DataCollectionPDU	/simnet/common/include/protocol/p_data.h
Originator	pointer to SimulationAddress	/simnet/common/include/protocol/address.h
Trans_ID	TransactionIdentifier	/simnet/common/include/protocol/address.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sPDU	pointer to StatusQueryVariant	/simnet/common/include/protocol/p_data.h
dPDU	DataCollectionPDU	/simnet/common/include/protocol/p_data.h
responsePDU	pointer to StatusResponseVariant	/simnet/common/include/protocol/p_data.h
dSize	int	Standard C type.
status	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_ASSOC_SNDDGRAM	Cannot access the network.	
Calls		
Function	**Where Described**	
Do_Generic_StatusQuery	Section 2.1.7.1.9.	
AssocSendResponse	Section 2.20.1.4.2.	
AssocError	Section 2.20.1.10.1.	
Called By		
Function	**Where Described**	
ProcessDatColTransaction	Section 2.1.6.2.3.	

Table 2.1-120: ProcessStatusQueryTransaction Information.

2.1.7.2 vtimeout.c

/simnet/mcc/Mother/vtimeout.c

vtimeout.c contains a routine to timeout vehicle appearance packets.

2.1.7.2.1 VTimeout_Appearance_Timeout

VTimeout_Appearance_Timeout iterates through all of the vehicle appearance packets and checks to see if any of them have timed out. Since the appearance packets are in a linked list, an appearance packet can not be deallocated until the next one is fetched. If an appearance packet is to be deallocated, it is first checked to see if it belongs to one of our simulators. If it does, a copy is kept to allow for reconstitution. The function call is VTimeout_Appearance_Timeout(). Table 2.1-121 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vlist_ptr	pointer to FQueue_t	/simnet/mcc/libmcc/fqueue.h
deletePDU	register pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
vaPDU	register pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
VehicleIDToSim()	function that returns a pointer to SimulatorStatus	See vehicle.c in Section 2.1.1.5.

Return Values		
Return Value	**Type**	**Meaning**
protocolVehicleDisappearanceTime_1000	int	

Calls	
Function	**Where Described**
FQueue_Scan	Section 2.21.1.13.6.
VehicleIDToSim	Section 2.1.1.5.3.
FQueue_Remove	Section 2.21.1.13.5.

Table 2.1-121: VTimeout_Appearance_Timeout Information.

2.1.8 Minefield Simulation

If a minefield has been emplaced by this MCC, this CSC is responsible for periodically sending out MarkerPDU's which describe where minefield flags are located. It also sends out MinefieldPDU's that describe where a minefield is located for data collection purposes. In addition, it listens periodically to see if a ground vehicle has crossed into a minefield and should be blown up. It issues an impact PDU if this is the case. It also listens to BreachedLanePDUs to update any cleared areas which might intersect this MCC's minefields. The following CSUs form its structure:

geometry.c
marker.c
minefield.c

2.1.8.1 geometry.c

/simnet/mcc/Mother/geometry.c

geometry.c contains routines for basic geometry utilities.

2.1.8.1.1 line_crosses_rect

line_crosses_rect determines if a line segment crosses a given rectangle. *pos* and *dest* are pointers to two floats which represent two (x,y) coordinate pairs. *rect_ptr* is a pointer to a rectangle or four (x,y) coordinate pairs. The line segment crossing the rectangle is defined by a line going from *pos* to *dest*. The function call is line_crosses_rect(pos, dest, rect_ptr). Table 2.1-122 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
pos	pointer to REAL	/simnet/common/include/global/sim_types.h.
dest	pointer to REAL	/simnet/common/include/global/sim_types.h.
rect_ptr	pointer to RECT	/simnet/mcc/Mother/geometry.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
slope	REAL	/simnet/common/include/global/sim_types.h.
dx	REAL	/simnet/common/include/global/sim_types.h.
dy	REAL	/simnet/common/include/global/sim_types.h.
y_int	REAL	/simnet/common/include/global/sim_types.h.
l_int	REAL	/simnet/common/include/global/sim_types.h.
t_int	REAL	/simnet/common/include/global/sim_types.h.
r_int	REAL	/simnet/common/include/global/sim_types.h.
b_int	REAL	/simnet/common/include/global/sim_types.h.
left	REAL	/simnet/common/include/global/sim_types.h.
top	REAL	/simnet/common/include/global/sim_types.h.
right	REAL	/simnet/common/include/global/sim_types.h.
bottom	REAL	/simnet/common/include/global/sim_types.h.
path_rect	RECT	/simnet/mcc/Mother/geometry.h

Return Values		
Return Value	**Type**	**Meaning**
TRUE	int	Line intersects rectangle.
FALSE	int	Line does not intersect rectangle.

Calls	
Function	**Where Described**
intersect_rect	Section 2.1.8.1.2.

Table 2.1-122: line_crosses_rect Information.

2.1.8.1.2 intersect_rect

intersect_rect determines if two rectangles, *r1* and *r2*, intersect. The function call is intersect_rect(r1, r2). Table 2.1-123 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
r1	pointer to RECT	/simnet/mcc/Mother/geometry.h
r2	pointer to RECT	/simnet/mcc/Mother/geometry.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
intersect	RECT	/simnet/mcc/Mother/geometry .h
Return Values		
Return Value	**Type**	**Meaning**
TRUE	int	Rectangles intersect.
FALSE	int	Rectangles don't intersect.
Called By		
Function	**Where Described**	
line_crosses_rect	Section 2.1.8.1.1.	

Table 2.1-123: intersect_rect Information.

2.1.8.1.3 init_rect

init_rect initializes a rectangle, specified by *rect_ptr*. The function call is init_rect(rect_ptr). Table 2.1-124 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
rect_ptr	pointer to RECT	/simnet/mcc/Mother/geometry .h

Table 2.1-124: init_rect Information.

2.1.8.1.4 inflate_rect

inflate_rect expands a rectangle, *rect_ptr*, by *amount* in all directions. The function call is inflate_rect(rect_ptr, amount). Table 2.1-125 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
rect_ptr	pointer to RECT	/simnet/mcc/Mother/geometry .h
amount	REAL	/simnet/common/include/global/sim_types.h.

Table 2.1-125: inflate_rect Information.

2.1.8.1.5 make_polygon

make_polygon creates a polygon. *num_vertices* is the number of vertices in the polygon to be created. *array_rep* is a list of vertices which will be in the polygon. *poly_rep* the polygon structure created and passed back to the user. The function call is make_polygon(num_vertices, array_rep, poly_rep). Table 2.1-126 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
num_vertices	int	Standard C type.
array_rep[MAXPOLYGON_VERTICES][2]	register matrix of REAL	/simnet/common/include/global/sim_types.h.
poly_rep	register pointer to POLYGON	/simnet/mcc/Mother/geometry.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
x	register REAL	/simnet/common/include/global/sim_types.h.
y	register REAL	/simnet/common/include/global/sim_types.h.
left	REAL	/simnet/common/include/global/sim_types.h.
right	REAL	/simnet/common/include/global/sim_types.h.
top	REAL	/simnet/common/include/global/sim_types.h.
bottom	REAL	/simnet/common/include/global/sim_types.h.
Called By		
Function	**Where Described**	
Emplace_Generic	Section 2.1.8.3.6.	

Table 2.1-126: make_polygon Information.

2.1.8.1.6 point_in_polygon

point_in_polygon determines if *point* is in *polygon*. The function call is point_in_polygon(point, polygon). Table 2.1-127 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
point	register pointer to REAL	/simnet/common/include/global/sim_types.h.
polygon	register pointer to POLYGON	/simnet/mcc/Mother/geometry.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register int	Standard C type.
num_crossed	int	Standard C type.
x0	register int	Standard C type.
y0	register int	Standard C type.
x1	register int	Standard C type.
y1	register int	Standard C type.
m	register REAL	/simnet/common/include/global/sim_types.h.
b	register REAL	/simnet/common/include/global/sim_types.h.
Return Values		
Return Value	**Type**	**Meaning**
FALSE	int	Point not in polygon.
num_crossed	int	Number of polygon line segment crossed going from point to infinity. An odd number means that we are within the polygon.
Calls		
Function	**Where Described**	
point_in_rect	Macro defined in /simnet/mcc/Mother/geometry.h.	
Called By		
Function	**Where Described**	
Mines_Point_In_Minefield	Section 2.1.8.3.13.	

Table 2.1-127: point_in_polygon Information.

2.1.8.2 marker.c

/simnet/mcc/Mother/marker.c

marker.c handles the maintainance and processing of minefield markers. Table 2.1-128 shows some of the variables used by marker.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
networkInterface	extern int	/simnet/mcc/include/MCC_net.h
markerLineTable	array MAX_MARKER_LINES of MarkerLine	/simnet/mcc/Mother/marker.c
maBuf	SimulationPDU	/simnet/common/include/protocol/p_sim.h
currentBatch	int	Standard C type.
currentPDU	int	Standard C type.
currentLine	int	Standard C type.
currentMarker	int	Standard C type.
retryPDUCount	int	Standard C type.
markerInPDU	int	Standard C type.
nextLineIndex	int	Standard C type.
nextMarkerID	int	Standard C type.

Table 2.1-128: marker.c Variable Information.

2.1.8.2.1 Marker_Emplace_Minefield

Marker_Emplace_Minefield creates the required lines of minefield markers which should be used to mark that minefield from the passed Minefield emplacement message. *vertices* is a pointer to a list of (x,y) coordinate pairs which defined the edges of a minefield somewhere on the terrain. *vertex_count* is the number of vertices pointed to by *vertices*. The function call is Marker_Emplace_Minefield(vertices, vertex_count). Table 2.1-129 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vertices	pointer to LongPt	/simnet/common/include/global/longpt.h
vertex_count	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
startPt	int	Standard C type.
endPt	int	Standard C type.
dx	double	Standard C type.
dy	double	Standard C type.
marker_azimuth	double	Standard C type.
Calls		
Function	**Where Described**	
radians_to_simnet_angle	libapp.h	
Marker_Add_Line	Section 2.1.8.2.2.	
Called By		
Function	**Where Described**	
EmplaceRequest	Section 2.1.1.2.2.	

Table 2.1-129: Marker_Emplace_Minefield Information.

2.1.8.2.2 Marker_Add_Line

Marker_Add_Line creates a line of markers. *type* determines the type of the marker, either MINEFIELD_MARKER or BREACH_MARKER. *spacing* is the amount of space, in meters, between the marker flags. *start* is the (x,y) coordinate defining the start of the line of flags and *end* is the coordinate pair for the end of the line of flags. *marker_azimuth* defines the angle of the marker flag ins SIMNET BAM (Binary Angle Measure) units from North (0). The function call is Marker_Add_Line(type, spacing, start, end, marker_azimuth). Table 2.1-130 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
spacing	int	Standard C type.
start	pointer to LongPt	/simnet/common/include/global/longpt.h
end	pointer to LongPt	/simnet/common/include/global/longpt.h
marker_azimuth	Angle	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
markerLine	pointer to MarkerLine	/simnet/mcc/Mother/marker.c
diffX	long	Standard C type.
diffY	long	Standard C type.
deltaX	int	Standard C type.
deltaY	int	Standard C type.
length	int	Standard C type.
potentialMarkers	int	Standard C type.
testLocation	array 2 of float	Standard C type.
testMarker	int	Standard C type.
realMarker	int	Standard C type.
Calls		
Function	**Where Described**	
Marker_Get_Next_Marker	Section 2.1.8.2.4.	
SQR	Macro defined in /simnet/mcc/Mother/marker.c.	
Marker_Point_In_Terrain	Section 2.1.8.2.5.	
tdb_get_z	Section 2.21.7.16.2.	
tdb_error	Section 2.21.7.17.1.	
Called By		
Function	**Where Described**	
BreachRequest	Section 2.1.1.2.3.	
Marker_Emplace_Minefield	Section 2.1.8.2.1.	

Table 2.1-130: Marker_Add_Line Information.

2.1.8.2.3 Marker_Broadcast_Markers

Marker_Broadcast_Markers broadcasts Marker "Appearance" PDUs for the maintained markers. It returns the interval to wait before calling the procedure again. The function call is Marker_Broadcast_Markers(). Table 2.1-131 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
maPDU	pointer to MarkerVariant	/simnet/common/include/protocol/p_sim.h
temp	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
temp	int	Period to wait before calling procedure again.
markerBatchPeriod	int	Constant defined in /simnet/mcc/include/MCC_timers.h. Period to wait before calling procedure again.
Calls		
Function	**Where Described**	
AssocSendDatagram	Section 2.20.1.2.1.	
PRO_SIM_MARKER_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h	
AssocError	Section 2.20.1.10.1.	
MAX	Macro defined in /simnet/mcc/Mother/marker.c.	

Table 2.1-131: Marker_Broadcast_Markers Information.

2.1.8.2.4 Marker_Get_Next_Marker

Marker_Get_Next_Marker returns a pointer to an available MarkerLine structure. The function call is Marker_Get_Next_Marker(). Table 2.1-132 describes the return values generated by this function.

Return Values		
Return Value	**Type**	**Meaning**
0	pointer to MarkerLine	No MarkerLine structure available.
&markerLineTable[nextLineIndex++]	pointer to MarkerLine	Next MarkerLine structure available.
Called By		
Function	**Where Described**	
Marker_Add_Line	Section 2.1.8.2.2.	

Table 2.1-132: Marker_Get_Next_Marker Information.

2.1.8.2.5 Marker_Point_In_Terrain

Marker_Point_In_Terrain tests to see if *location* is on the terrain. The function call is Marker_Point_In_Terrain(location). Table 2.1-133 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
location	pointer to float	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Point not on terrain.
1	int	Point on terrain.
Called By		
Function	**Where Described**	
Marker_Add_Line	Section 2.1.8.2.2.	

Table 2.1-133: Marker_Point_In_Terrain Information.

2.1.8.3 minefield.c

/simnet/mcc/Mother/minefield.c

minefield.c does the processing to check for simulators blowing up in minefields. Currently, only ground vehicles are affected. Table 2.1-134 shows some variables used by minefield.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
Minefields	array MAX_MINEFIELDS of Minefield_t	/simnet/mcc/Mother/minefield.h
Minefield_Object_Counter	unsigned short	Standard C type.
Minefield_Event	unsigned long	Standard C type.
VAPDU_Ptr	pointer to FQueue_t	/simnet/mcc/libmcc/fqueue.h
networkInterface	extern int	Standard C type.

Table 2.1-134: minefield.c Variable Information.

2.1.8.3.1 Current_Minefield_Count

Current_Minefield_Count returns the number of minefields currently in use. The function call is Current_Minefield_Count(). Table 2.1-135 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register int	Standard C type.
mcount	register int	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
mcount	unsigned char	Number of minefields in use.
Calls		
Function	**Where Described**	
Mines_Max_Minefields	Section 2.1.8.3.4.	
Called By		
Function	**Where Described**	
Emplace_Generic	Section 2.1.8.3.6.	
Mines_Point_In_Minefield	Section 2.1.8.3.13.	

Table 2.1-135: Current_Minefield_Count Information.

2.1.8.3.2 Alloc_Minefield

Alloc_Minefield allocates a pointer to an available Minefield structure. The function call is Alloc_Minefield(). Table 2.1-136 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
NULL	pointer to Minefield_t	No minefields available.
&Minefields[i]	pointer to Minefield_t	Next available minefield.
Calls		
Function	**Where Described**	
Mines_Max_Minefields	Section 2.1.8.3.4.	
Called By		
Function	**Where Described**	
Emplace_Generic	Section 2.1.8.3.6.	

Table 2.1-136: Alloc_Minefield Information.

2.1.8.3.3 Free_Minefield

Free_Minefield frees a minefield, *Minefield* from use. The function call is Free_Minefield(Minefield). Table 2.1-137 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h

Return Values		
Return Value	**Type**	**Meaning**
0	short	Always returned.

Table 2.1-137: Free_Minefield Information.

2.1.8.3.4 Mines_Max_Minefields

Mines_Max_Minefields returns the maximum number of minefields allowed. The function call is Mines_Max_Minefields(). Table 2.1-138 describes the return value generated by this function.

Return Values		
Return Value	**Type**	**Meaning**
MAX_MINEFIELDS	unsigned char	Maximum number of minefields allowed. MAX_MINEFIELDS is a constant defined in minefield.c
Called By		
Function	**Where Described**	
Current_Minefield_Count	Section 2.1.8.3.1.	
Alloc_Minefield	Section 2.1.8.3.2.	
Emplace_Generic	Section 2.1.8.3.6.	
Mines_Process_Breach_Datagram	Section 2.1.8.3.10.	

Table 2.1-138: Mines_Max_Minefields Information.

2.1.8.3.5 Mines_Max_Vertices

Mines_Max_Vertices returns the maximum number of vertices for a minefield. The function call is Mines_Max_Vertices(). Table 2.1-139 describes the return value generated by this function.

Return Values		
Return Value	**Type**	**Meaning**
MAX_VERTICES	unsigned char	Maximum number of vertices allowed. MAX_VERTICES is a constant defined in minefield.c
Called By		
Function	**Where Described**	
Make_Minefield_Status	Section 2.1.8.3.17.	

Table 2.1-139: Mines_Max_Vertices Information.

2.1.8.3.6 Emplace_Generic

Emplace_Generic creates a minefield structure. *Vertices* is a list of (x,y) coordinate pairs defining the edges of the minefield/breach lane to be emplaced. *Vertex_Count* is the number of coordinate pairs in *Vertices*. *Type* describes whether the area is a breach lane or a minefield. The function call is Emplace_Generic(Vertices, Vertex_Count, Type). Table 2.1-140 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Vertices	array of LongPt	/simnet/common/include/global/longpt.h
Vertex_Count	unsigned char	Standard C type.
Type	u_short	
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h
v_array	11 by 2 matrix of double	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
NULL	pointer to Minefield_t	Minefield not emplaced.
minefield	pointer to Minefield_t	Minefield emplaced.
Calls		
Function	**Where Described**	
Current_Minefield_Count	Section 2.1.8.3.1.	
Mines_Max_Minefields	Section 2.1.8.3.4.	
Alloc_Minefield	Section 2.1.8.3.2.	
make_polygon	Section 2.1.8.1.5.	
Called By		
Function	**Where Described**	
Mines_Emplace_Minefield	Section 2.1.8.3.7.	
Mines_Breach_Lane	Section 2.1.8.3.8.	

Table 2.1-140: Emplace_Generic Information.

2.1.8.3.7 Mines_Emplace_Minefield

Mines_Emplace_Minefield creates a minefield structure by calling Emplace_Generic and assigns mine densities. *Vertices* is a list of (x,y) coordinate pairs defining the edges of the minefield lane to be emplaced. *Vertex_Count* is the number of coordinate pairs in *Vertices*. *ATDensity* is the density of Anti_Tank Mines, *APFDensity* is the density of Anti_Personnel fragmenting mines and *APBDensity* is the density of Anti_Personnel blast mines. All mine densities are given in mines per square meter. The function call is Mines_Emplace_Minefield(Vertices, Vertex_Count, ATDensity, APFDensity,

APBDensity). Table 2.1-141 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Vertices	array of LongPt	/simnet/common/include/global/longpt.h
Vertex_Count	unsigned char	Standard C type.
ATDensity	float	Standard C type.
APFDensity	float	Standard C type.
APBDensity	float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h
Return Values		
Return Value	**Type**	**Meaning**
-1	int	Minefield not emplaced.
0	int	Minefield emplaced.
Calls		
Function	**Where Described**	
Emplace_Generic	Section 2.1.8.3.6.	
Called By		
Function	**Where Described**	
EmplaceRequest	Section 2.1.1.2.2.	

Table 2.1-141: Mines_Emplace_Minefield Information.

2.1.8.3.8 Mines_Breach_Lane

Mines_Breach_Lane allocates a breached lane. *Vertices* is a list of (x,y) coordinate pairs defining the edges of the breach lane. *Vertex_Count* is the number of coordinate pairs in *Vertices*. If *Breach_ID* is NULL, then the ID that was allocated by Emplace_Generic is left in. Otherwise the ID that was passed in is used; in this case, the ID is from some other system which just breached an area and notified us over the network. The function call is Mines_Breach_Lane(Vertices, Vertex_Count, Breach_ID). Table 2.1-142 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Vertices	array of LongPt	/simnet/common/include/global/longpt.h
Vertex_Count	unsigned char	Standard C type.
Breach_ID	ObjectID	/simnet/common/include/protocol/basic.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h
Return Values		
Return Value	**Type**	**Meaning**
NULL	pointer to Minefield_t	No breach lane emplaced.
minefield	pointer to Minefield_t	Breach lane emplaced.
Calls		
Function	**Where Described**	
Emplace_Generic	Section 2.1.8.3.6.	
Called By		
Function	**Where Described**	
Mines_Emplace_Breach	Section 2.1.8.3.9.	
Mines_Process_Breach_Datagram	Section 2.1.8.3.10.	

Table 2.1-142: Mines_Breach_Lane Information.

2.1.8.3.9 Mines_Emplace_Breach

Mines_Emplace_Breach emplaces a breach lane into some area on the terrain database and indicates to the rest of the network that this has occured. *Vertices* is a list of (x,y) coordinate pairs defining the edges of the breach lane to be emplaced and *Vertex_count* is the number of coordinate pairs in *Vertices*. The function call is Mines_Emplace_Breach(Vertices, Vertex_Count). Table 2.1-143 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Vertices	array of LongPt	/simnet/common/include/global/longpt.h
Vertex_Count	unsigned char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h
simPDU	SimulationPDU	/simnet/common/include/protocol/p_sim.h
breach	pointer to BreachedLaneVariant	/simnet/common/include/protocol/p_sim.h
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
-1	int	Unsuccessful.
0	int	Successful.

Calls	
Function	**Where Described**
Mines_Breach_Lane	Section 2.1.8.3.8.
AssocSendDatagram	Section 2.20.1.2.1.
AssocError	Section 2.20.1.10.1.
Called By	
Function	**Where Described**
BreachRequest	Section 2.1.1.2.3.

Table 2.1-143: Mines_Emplace_Breach Information

2.1.8.3.10 Mines_Process_Breach_Datagram

Mines_Process_Breach_Datagram is called when a breach lane status packet, *simPDU*, is received. If this breach lane is not yet recorded, it will be recorded at this time. The function call is Mines_Process_Breach_Datagram(simPDU). Table 2.1-144 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
simPDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
breach	pointer to BreachedLaneVariant	/simnet/common/include/protocol/p_sim.h
Vertices	array MAX_VERTICES of LongPt	/simnet/common/include/global/longpt.h
i	int	Standard C type.
Vertex_Count	int	Standard C type.
Calls		
Function	**Where Described**	
Mines_Max_Minefields	Section 2.1.8.3.4.	
Mines_Breach_Lane	Section 2.1.8.3.8.	
Called By		
Function	**Where Described**	
ProcessSimDatagram	Section 2.1.6.2.9.	

Table 2.1-144: Mines_Process_Breach_Datagram Information.

2.1.8.3.11 Mines_Get_Info

Mines_Get_Info allows queries to be made about what minefields and breach lanes are currently emplaced. This is needed for the MCC console to be able to restart after it has crashed. This routine should be invoked iteratively in order to get information on subsequent minefields. *Context* contains the number of minefields last used when this routine was called. The first time the routine is called, this should be zero. *Vertices* is a list of (x.y) coordinate pairs defining the edges of the next minefield lane. *Vertex Count* is the

number of coordinate pairs in *Vertices*. *ATDensity* is the density of Anti_Tank Mines, *APFDensity* is the density of Anti_Personnel fragmenting mines and *APBDensity* is the density of Anti_Personnel blast mines. All mine densities are given in mines per square meter. The function call is Mines_Get_Info(Context, Vertices, Vertex_Count, ATDensity, APFDensity, APBDensity). Table 2.1-145 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Context	pointer to Minefield_Context_t	/simnet/mcc/Mother/minefield.h
Vertices	array of LongPt	/simnet/common/include/global/longpt.h
Vertex_Count	unsigned char	Standard C type.
ATDensity	pointer to float	Standard C type.
APFDensity	pointer to float	Standard C type.
APBDensity	pointer to float	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	No active minefields.
-1	int	Successful.

Table 2.1-145: Mines_Get_Info Information.

2.1.8.3.12 Blow_Vehicle

Blow_Vehicle creates a SimulationPDU, *vaPDU*, when a vehicle is blown up by a mine, determined by *minefield* and *munition*. The function call is Blow_Vehicle(vaPDU, minefield, munition). Table 2.1-146 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vaPDU	pointer to VehicleAppearanceVaraint	/simnet/common/include/protocol/p_sim.h
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h
munition	ObjectType	/simnet/common/include/protocol/basic.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
if_buf	SimulationPDU	/simnet/common/include/protocol/p_sim.h
if_pdu	pointer to IndirectFireVariant	/simnet/common/include/protocol/p_sim.h
detonation	pointer to IndirectFireDetonation	/simnet/common/include/protocol/p_sim.h
status	int	Standard C type.

Calls	
Function	**Where Described**
AssocSendDatagram	Section 2.20.1.2.1.
PRO_SIM_IND_FIRE_SIZE	Macro defined in /simnet/common/include/protocol/p_size.h
AssocError	Section 2.20.1.10.1.

Called By	
Function	**Where Described**
Crossing_Minefield	Section 2.1.8.3.14.

Table 2.1-146: Blow_Vehicle Information.

2.1.8.3.13 Mines_Point_In_Minefield

Mines_Point_In_Minefield determines if a point, specified by *x* and *y*, is in a minefield. Note that the minefields are searched in reverse chronological order. This algorithm deals correctly with laying down multiple minefields and breach lanes which cover the same geographical area. The function call is Mines_Point_In_Minefield(x, y). *Table 2.1-147* describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
x	double	Standard C type.
y	double	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
xed	int	Standard C type.
i	int	Standard C type.
minefield_count	int	Standard C type.
location	array 2 of double	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
-1	int	Point not in minefield or in breached lane.
i	int	Point in minefield.

Calls	
Function	**Where Described**
Current_Minefield_Count	Section 2.1.8.3.1.
point_in_polygon	Section 2.1.8.1.6.

Called By	
Function	**Where Described**
Mines_Check_Minefields	Section 2.1.8.3.15.
FindAnOpenSpot	Section 2.1.2.5.1.

Table 2.1-147: Mines_Point_In_Minefield Information.

2.1.8.3.14 Crossing_Minefield

Crossing_Minefield determines if a vehicle should be blown up given that it is in a minefield. The chance of a vehicle being blown up is determined by the number of mines that the vehicle has crossed since the last scan. This is calculated by knowing the mine density and assuming that the vehicle is moving at a constant velocity. There is a 66.7% chance that the vehicle has exploded for each mine it has crossed. There is also a chance equal to the density of the minefield that the vehicle hit a mine if it did not pass over any "probabilistic" mines. Otherwise, we would not blow up vehicles that moved very slowly. *vaPDU* is the vehicle appearance PDU of the vehicle which is crossing the minefield. *minefield* is a pointer to the minefield structure containing information on the density and types of mines in the minefield. *vehicle_domain* is the vehicle domain bitmask which is used to tell if the vehicle is affected by the minefield. The function call is Crossing_Minefield(vaPDU, minefield, vehicle_domain). Table 2.1-148 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vaPDU	pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h
vehicle_domain	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
area_covered	float	Standard C type.
mines_crossed	unsigned long	Standard C type.
i	unsigned int	Standard C type.
velocity	unsigned int	Standard C type.
delta_t	float	Standard C type.
delta_x	double	Standard C type.
delta_y	double	Standard C type.
delta_z	double	Standard C type.
Calls		
Function	**Where Described**	
EXP2	Macro defined in /simnet/mcc/Mother/minefield.c.	
Blow_Vehicle	Section 2.1.8.3.12.	

Called By	
Function	**Where Described**
Mines_Check_Minefields	Section 2.1.8.3.15.

Table 2.1.148: Crossing_Minefield Information.

2.1.8.3.15 Mines_Check_Minefields

Mines_Check_Minefields checks to see which vehicles are in minefields. If a vehicle is in a minefield, then further checking is done to see if it is blown up. On each scan through this routine, we only check 1/3 of the vehicles unless there are less than PACKET_MINIMUM (constant defined in minefield.c) vehicles. In that case, all the vehicles are checked. The function call is Mines_Check_Minefields(). Table 2.1-149 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vlist_ptr	pointer to FQueue_t	/simnet/mcc/libmcc/fqueue.h
packets_inspected	register int	Standard C type.
inspect_max	register int	Standard C type.
domain	int	Standard C type.
enviroment	int	Standard C type.
i	int	Standard C type.
vaPDU	register pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
Return Values		
Return Value	**Type**	**Meaning**
mcc->mineScanInterval	int	Time delay before calling procedure again.
Calls		
Function	**Where Described**	
FQueue_Scan	Section 2.21.1.13.6.	
Mines_Point_In_Minefield	Section 2.1.8.3.13.	
Crossing_Minefield	Section 2.1.8.3.14.	

Table 2.1-149: Mines_Check_Minefields Information.

2.1.8.3.16 Make_Breach_Status

Make_Breach_Status creates a SimulationPDU, *simPDU*, with information about a breached lane in a minefield, *minefield*. The function call is Make_Breach_Status(simPDU, minefield). Table 2.1-150 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
simPDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h
minefield	pointer to Minefield_t	/simnet/mcc/Mother/minefield.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
br_pdu	pointer to BreachedLaneVariant	/simnet/common/include/protocol/p_sim.h
j	int	Standard C type.
Called By		
Function	**Where Described**	
Mines_Send_Minefield_Status	Section 2.1.8.3.18.	

Table 2.1-150: Make_Breach_Status Information.

2.1.8.3.17 Make_Minefield_Status

Make_Minefield_Status creates a simulationPDU, *simPDU*, with information about a minefield, *minefield*. The function call is Make_Minefield_Status(simPDU, minefield). Table 2.1-151 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
simPDU	pointer to SimulationPDU	/simnet/common/include/protocol/p_sim.h
minefield	pointer to Minefield_t	/simnet/incc/Mother/minefield.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
mf_pdu	pointer to MinefieldVariant	/simnet/common/include/protocol/p_sim.h
j	int	Standard C type.
Calls		
Function	**Where Described**	
Mines_Max_Vertices	Section 2.1.8.3.5.	
Called By		
Function	**Where Described**	
Mines_Send_Minefield_Status	Section 2.1.8.3.18.	

Table 2.1-151: Make_Minefield_Status Information.

2.1.8.3.18 Mines_Send_Minefield_Status

Mines_Send_Minefield_Status sends out a status packet describing each minefield and breach lane which this system created. The function call is Mines_Send_Minefield_Status(). Table 2.1-152 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
mf_buf	SimulationPDU	/simnet/common/include/protocol/p_sim.h
i	int	Standard C type.
status	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
30000	int	
Calls		
Function	**Where Described**	
Make_Minefield_Status	Section 2.1.8.3.17.	
Make_Breach_Status	Section 2.1.8.3.16.	
AssocSendDatagram	Section 2.20.1.2.1.	
AssocError	Section 2.20.1.10.1.	

Table 2.1-152: Mines_Send_Minefield_Status Information.

2.2 The SCC (SIMNET Control Console) Process

The SCC host process communicates directly with the SCC Macintosh application. Its functions include initializing components of a simulation exercise, placing combat vehicles, reconstituting vehicles, and placing static vehicles. Its structure is shown in Figure 2.2-1 below.

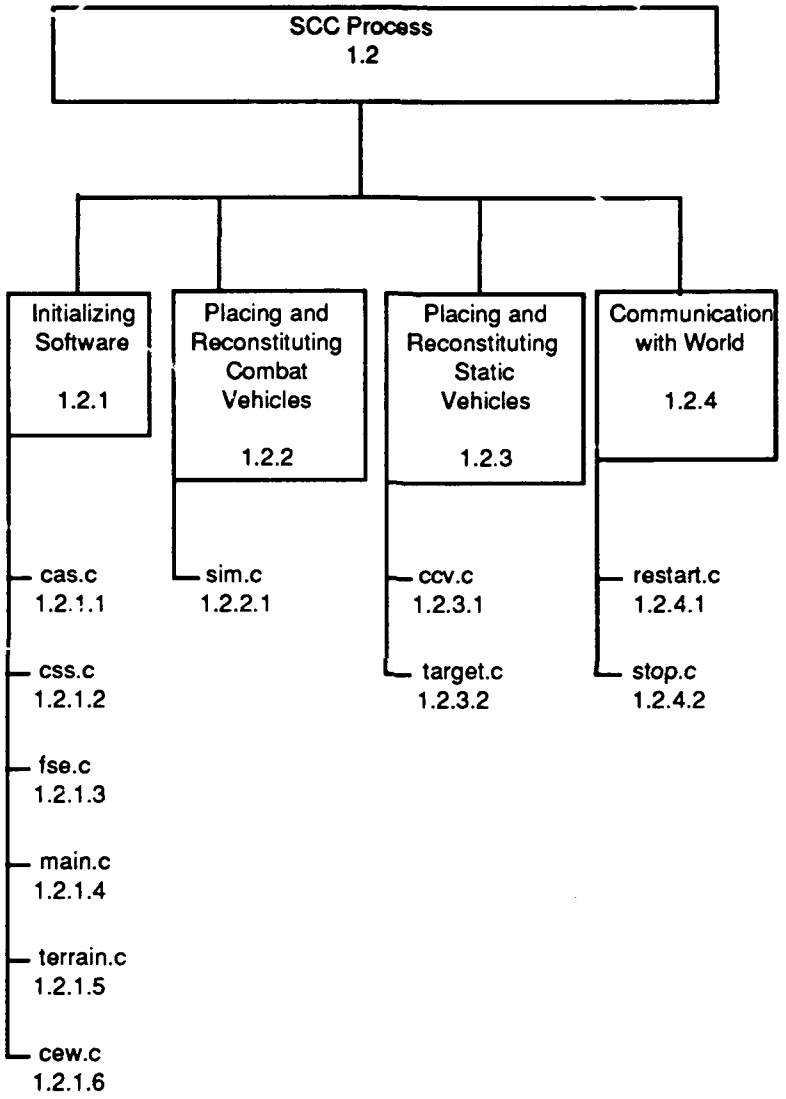


Figure 2.2-1: SCC (SIMNET Control Console) Process Structure.

The SCC process sends and receives the interprocess messages listed in Table 2.2-1.

Interprocess Message	Received From	Sent To	In Response To	Data Sent	Transaction Type
InitTerrain		Mother	Exercise initialization complete	-terrain database	One-way interaction
ShowCCV		Mother	CCV needs to be visible	-ownership -identifier -type -alignment -role -company -location -direction	First part of two-way interaction
HideCCV		Mother	CCV needs to be invisible	- indentifier	One-way interaction
KilledCCV	Mother		When CCV killed as shown by these PDUs: VehicleImpact Collision IndirectFire	- identifier	One-way interaction
SimAllocated		Place	Combat vehicle simulator allocated to company at SCC Console	- identifier - company	One-way interaction
SimPlaced		Place	Combat vehicle simulator placed at SCC Console	- identifier	One-way interaction
SimProblem	Mother		If combat vehicle simulator - didn't respond to ActivatePDU, - not sending Vehicle Appearance PDUs after sending Activating PDU, - stopped sending Vehicle Appearance PDUs.	- identifier - problem	One-way interaction
ActivateVehicle		Mother	Activate or reconstitute combat vehicle simulator	- identifier - reason - type - location - alignment - bumper # - hull and turret orientation - supplies and failure status	First part of two-way interaction

ActivateResult	Mother		Placed combat vehicle or CCV.	- identifier - result	Second part of two-way interaction (either ShowCCV or ActivateVehicle)
ArtyDispatch		FSE	BattleMaster wants to dispatch platoon of howitzers	- identifier - platoon	One-way interaction
ArtyArrive		FSE	Howitzers have arrived	- identifier - platoon - location - azimuth	One-way interaction
Arty Reconstitute		FSE	BattleMaster wants to reconstitute platoon of howitzer battery	- identifier - platoon - location - supply quantities	One-way interaction
Sorties		CAS	BattleMaster has allocated sorties	- number of sorties - number of sorties that may be preplanned	One-way interaction
DepotDisplace		Admin Maint	BattleMaster has displaced ammunition or fuel depots		One-way Interaction
Truck Reconstitute		Admin Maint	Supply truck or maintenance team reconstituted	- identifier - type - location - supplies - company - force	One-way interaction

Table 2.2-1: SCC Process Interprocess Messages

It is broken down into the following second level CSUs:

Initialization
Placing Combat Vehicles
Placing Static Vehicles
Communication with World

2.2.1 Initialization

The SCC first opens a connection so that it can communicate with the SCC Macintosh. The SCC spawns the other console processes as it receives the corresponding messages from the SCC Macintosh. It sends all of the available terrain information to the SCC Mac so the user can choose which database to use to fight the battle. In addition, the SCC Mac allows a "Battlemaster", who is equipped with the correct password, to reconstitute vehicles as the battle progresses. The SCC process receives messages to reconstitute vehicles and sends a message to the Mother process to either change the location of the CCV or deactivate and re-activate a combat vehicle simulator. Any of the CCVs or the

MCC's combat vehicles may be reconstituted. The Battlemaster may also change the locations of any of the depots. The new locations are sent to the SCC to update its data. The Initialization CSC is composed of the following CSUs:

cas.c
css.c
fse.c
main.c
terrain.c
cew.c

2.2.1.1 cas.c

/simnet/mcc/SCC/cas.c

cas.c contains routines that implement theSCC support for the CloseAirSupport Simulation.

2.2.1.1.1 ProcessCASRequest

ProcessCASRequest processes a request, *t*, from the Macintosh to start the CAS console. The function call is ProcessCASRequest(t). Table 2.2-2 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCCASRequest	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
LookupProcessNumber	Section 2.21.1.26.3.	
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
TurnRadioOn	Section 2.21.1.23.3.	
SpawnProcess	Section 2.21.1.26.1.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-2: ProcessCASRequest Information.

2.2.1.2 css.c

/simnet/mcc/SCC/css.c

css.c contains routines that implement the SCC support for Combat Service Support. It also handles some aspects of the reconstitution of vehicles.

2.2.1.2.1 ProcessTruckInitRequest

ProcessTruckInitRequest processes a request, *t*, from the Macintosh supplying truck initialization or reconstitution parameters. The function call is ProcessTruckInitRequest(t). Table 2.2-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	register pointer to SCCTruckInitRequest	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
LookupProcessNumber	Section 2.21.1.26.3.	
ReconstituteTruck	Section 2.2.1.2.3.	
InitializeTruck	Section 2.2.1.2.2.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-3: ProcessTruckInitRequest Information.

2.2.1.2.2 InitializeTruck

InitializeTruck records a truck's initial parameters, given by *req*, in shared memory. The function call is InitializeTruck(req). Table 2.2-4 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
req	register pointer to SCCTruckInitRequest	/simnet/mcc/include/SCC.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pars	register pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
organiazation	OrganizationalUnit	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h

Calls	
Function	**Where Described**
InitUnit	Section 2.21.1.19.4.
GetTruckParameters	Section 2.21.1.29.1.
Called By	
Function	**Where Described**
ProcessTruckInitRequest	Section 2.2.1.2.1.

Table 2.2-4: InitializeTruck Information.

2.2.1.2.3 ReconstituteTruck

ReconstituteTruck tells the Admin or Maint process to reconstitute a truck. *req* contains information which describes the truck's alignment, caspabilities and location. The function call is ReconstituteTruck(req). Table 2.2-5 describes the parameters used and functions called using this function. *req* holds information about a truck's alignment, capabilities, and location.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
req	register pointer to SCCTruckInitRequest	/simnet/mcc/include/SCC.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
process	int	Standard C type.
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
InitUnit	Section 2.21.1.19.4.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
ProcessTruckInitRequest	Section 2.2.1.2.1.	

Table 2.2-5: ReconstituteTruck Information.

2.2.1.2.4 ProcessUMCPDisplaceRequest

ProcessUMCPDisplaceRequest processes a request, *t*, from the SCC console to displace the UMCP. The function call is ProcessUMCPDisplaceRequest(t). Table 2.2-6 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCUMCPDisplaceRequest	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
LookupProcessNumber	Section 2.21.1.26.3.	
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-6: ProcessUMCPDisplaceRequest Information.

2.2.1.2.5 ProcessDepotRequest

ProcessDepotRequest processes a request, *t*, from the SCC console reporting the locations of ammo and fuel depots. The function call is ProcessDepotRequest(t). Table 2.2-7 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCDepotsRequest	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
LookupProcessNumber	Section 2.21.1.26.3.	
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-7: ProcessDepotRequest Information.

2.2.1.2.6 ProcessCSSRequest

ProcessCSSRequest processes a request, *t*, from the SCC console marking the completion of the CSS battalion initialization. The function call is ProcessCSSRequest(t). Table 2.2-8 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCCSSRequest	/simnet/mcc/include/SCC.h
i	int	Standard C type.
pt	LongPt	/simnet/common/include/global/longpt.h
unit	OrganizationalUnit	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
marking	VehicleMarking	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
GetGuises	Section 2.21.1.17.1.	
InitUnit	Section 2.21.1.19.4	
DisplayCCV	Section 2.21.1.6.1.	
TurnRadioOn	Section 2.21.1.23.3.	
SpawnProcess	Section 2.21.1.26.1.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-8: ProcessCSSRequest Information.

2.2.1.2.7 ProcessTruckQueryRequest

ProcessTruckQueryRequest processes a request, *t*, from the Macintosh for the current state of a truck. The function call is ProcessTruckQueryRequest(t). Table 2.2-9 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	register pointer to SCCTruckQueryRequest	/simnet/mcc/include/SCC.h
rsp	register pointer to SCCTruckQueryResponse	/simnet/mcc/include/SCC.h
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
unit	OrganizationalUnit	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
GetTruckParameters	Section 2.21.1.29.1.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-9: ProcessTruckQueryRequest Information.

2.2.1.3 fse.c

/simnet/mcc/SCC/fse.c

fse.c contains routines which implement the SCC process support to the Fire Support Element.

2.2.1.3.1 ProcessFSERequest

ProcessFSERequest processes a request, *t*, from the Macintosh to start the FSE console. The function call is ProcessFSERequest(t). Table 2.2-10 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCFSERequest	/simnet/mcc/include/SCC.h
i	short	Standard C type.
j	short	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
TurnRadioOn	Section 2.21.1.23.3.	
SpawnProcess	Section 2.21.1.26.1.	
SendConsoleResponse	Section 2.21.1.7.6.	

Called By	
Function	**Where Described**
RequestArrived	Section 2.2.1.4.3.

Table 2.2-10: ProcessFSERequest Information.

2.2.1.3.2 ProcessArtyQueryRequest

ProcessArtyQueryRequest processes a request, *t*, from the Macintosh to supply the latest status of the artillery. The function call is ProcessArtyQueryRequest(t). Table 2.2-11 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCArtyQueryRequest	/simnet/mcc/include/SCC.h
rsp	pointer to SCCArtyQueryResponse	/simnet/mcc/include/SCC.h
battery	short	Standard C type.
i	short	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-11: ProcessArtyQueryRequest Information.

2.2.1.3.3 ProcessArtyReconstRequest

ProcessArtyReconstRequest processes a request, *t*, from the Macintosh to reconstitute an artillery battery. The function call is ProcessArtyReconstRequest(t). Table 2.2-12 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCArtyReconstRequest	/simnet/mcc/include/SCC.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc .h
i	short	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-12: ProcessArtyReconstRequest Information.

2.2.1.3.4 ProcessArtyDispatchRequest

ProcessArtyDispatchRequest processes a request, *t*, from the Macintosh to displace a howitzer battery. The function call is ProcessArtyDispatchRequest(t). Table 2.2-13 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCArtyDispatchRequest	/simnet/mcc/include/SCC.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc .h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-13: ProcessArtyDispatchRequest Information.

2.2.1.3.5 ProcessArtyArriveRequest

ProcessArtyArriveRequest processes a request, *t*, from the Macintosh indicating which howitzer battery has arrived. The function call is ProcessArtyArriveRequest(t). Table 2.2-14 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCArtyArriveRequest	/simnet/mcc/include/SCC.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-14: ProcessArtyArriveRequest Information.

2.2.1.4 main.c

/simnet/mcc/SCC/main.c

main.c contains the host process that is a counterpart to the SIMNET Control Console Macintosh.

2.2.1.4.1 main

main is the entry point to the SCC process. The function call is main(argc, argv). Table 2.2-15 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.

Calls	
Function	**Where Described**
InitializeProcess	Section 2.21.1.19.1.
InitRadios	Section 2.21.1.23.1.
OpenHostSocket	Section 2.21.1.7.1.
ActivateConsole	Section 2.21.1.7.2.
AlarmsEnabled	Section 2.21.2.4.4.

Table 2.2-15: main Information.

2.2.1.4.2 ProcessMessage

ProcessMessage processes a message from another process. *type* describes the message as either one-way, a synchronous or asynchronous request or an asynchronous respones. *kind* describes the kind of message, *length* is the length of the data buffer and *data* is a pointer to an MCCMessageBuffer. The function call is ProcessMessage(type, kind, length, data). Table 2.2-16 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	long	Standard C type.
length	int	Standard C type.
data	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
alarmsEnabled	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESSAGE	Message type was unexpected.	
Calls		
Function	**Where Described**	
AlarmsEnabled	Section 2.21.2.4.4.	
ProcessExitMessage	Section 2.2.4.2.2.	
RestartConsole	Section 2.2.4.1.1.	
ResetClock	Section 2.2.1.4.4.	
BroadcastToChildren	Section 2.2.1.4.5.	
TargetKilled	Section 2.2.3.2.2.	
TurnRadioOff	Section 2.21.1.23.4.	
ReportSimProblem	Section 2.2.2.1.5.	
SimAllocedRemotely	Section 2.2.2.1.3.	
SimPlacedRemotely	Section 2.2.2.1.4.	

Table 2.2-16: ProcessMessage Information.

2.2.1.4.3 RequestArrived

RequestArrived processes an ATP Request, *t*, from the SCC Macintosh. The function call is RequestArrived(t). Table 2.2-17 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vehicle	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_GETREQUEST_FAILED	ATPGetRequest failed.	
MCC_UNKNOWN_REQUEST	Unknown request from Macintosh.	

Calls	
Function	**Where Described**
ATPGetRequest	Section 2.21.1.1.5.
ProcessTerrainRequest	Section 2.2.1.5.1.
ProcessMapSheetsRequest	Section 2.2.1.5.2.
ProcessExerciseRequest	Section 2.2.1.5.3.
ProcessGridInfoRequest	Section 2.21.1.16.1.
ProcessTOCRequest	Section 2.2.3.1.1.
ProcessALOCRequest	Section 2.2.3.1.3.
ProcessTruckInitRequest	Section 2.2.1.2.1.
ProcessTruckQueryRequest	Section 2.2.1.2.7.
ProcessCSSRequest	Section 2.2.1.2.6.
ProcessFSERequest	Section 2.2.1.3.1.
ProcessCASRequest	Section 2.2.1.1.1.
ProcessStopRequest	Section 2.2.4.2.1.
ProcessTargetSetRequest	Section 2.2.3.2.1.
ProcessArtyQueryRequest	Section 2.2.1.3.2.
ProcessArtyReconstRequest	Section 2.2.1.3.3.
ProcessDepotRequest	Section 2.2.1.2.5.
ProcessArtyDispatchRequest	Section 2.2.1.3.4.
ProcessArtyArriveRequest	Section 2.2.1.3.5.
ProcessTOCDispatchRequest	Section 2.2.3.1.2.
ProcessALOCDispatch Request	Section 2.2.3.1.4.
ProcessPermitOptions Request	Section 2.2.1.4.6.
ProcessSimExistsRequest	Section 2.21.1.25.1.
ProcessSimInitRequest	Section 2.21.1.25.2.
SimPlacedLocally	Section 2.2.2.1.2.
ProcessSimQueryRequest	Section 2.21.1.25.4.
ProcessSimAllocRequest	Section 2.2.2.1.1.
ProcessUMCPDisplace Request	Section 2.2.1.2.4.
ProcessCEWinitRequest	Section 2.2.1.6.1.
ProcessCEWReconst Request	Section 2.2.1.6.5.
ProcessCEWQueryRequest	Section 2.2.1.6.2.
ProcessCEWStartRequest	Section 2.2.1.6.3.

Table 2.2-17: RequestArrived Information.

2.2.1.4.4 ResetClock

ResetClock tells the Macintosh what time it is. The function call is ResetClock(). Table 2.2-18 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCClockRequest	/simnet/mcc/include/SCC.h

Calls	
Function	**Where Described**
MacintoshTime	Section 2.21.1.27.1.
ATPPut	Section 2.21.1.7.4.
Called By	
Function	**Where Described**
ProcessMessage	Section 2.2.1.4.2.

Table 2.2-18: ResetClock Information.

2.2.1.4.5 BroadcastToChildren

BroadcastToChildren sends a message to each process that has been spawned. *kind* is the kind of IPC message being sent. The message contains no data. The function call is BroadcastToChildren(kind). Table 2.2-19 describes the parameters used and functions called using this function. *kind* is the kind of IPC message being sent. The message has no data to it.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
kind	long	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	register int	Standard C type.
mypid	register int	Standard C type.
numberChildren	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
numberChildren	int	Number of children notified.
Calls		
Function	**Where Described**	
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.2.1.4.2.	
ShutdownChildren	Section 2.2.4.2.4.	

Table 2.2-19: BroadcastToChildren Information.

2.2.1.4.6 ProcessPermitOptionsRequest

ProcessPermitOptionsRequest processes a request, *t*, from the console as to what optional elements are permitted. The function call is ProcessPermitOptionsRequest(t). Table 2.2-20 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
rsp	pointer to SCCPermitOptionsResponse	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-20: ProcessPermitOptionsRequest Information.

2.2.1.5 terrain.c

/simnet/mcc/SCC/terrain.c

terrain.c contains routines which tell the Macintosh about the terrain database. Table 2.2-21 shows the variables used by terrain.c.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
terrains	array maxNumberTerrains of TDB_INFO	/simnet/common/libsrc/libtdb/tdb.h

Table 2.2-21: terrain.c Variable Information.

2.2.1.5.1 ProcessTerrainRequest

ProcessTerrainRequest processes a request, *t*, from the Macintosh for terrain information. The function call is ProcessTerrainRequest(t). Table 2.2-22 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCTerrainRequest	/simnet/mcc/include/SCC.h
rsp	pointer to SCCTerrainResponse	/simnet/mcc/include/SCC.h
Errors		
Error Name	**Reason for Error**	
MCC_NO_TERRAIN		

Calls	
Function	**Where Described**
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.
SendConsoleResponse	Section 2.21.1.7.6.
tdb_get_tdb_info	Section 2.21.7.27.3.
Called By	
Function	**Where Described**
RequestArrived	Section 2.2.1.4.3.

Table 2.2-22: ProcessTerrainRequest Information.

2.2.1.5.2 ProcessMapSheetsRequest

ProcessMapSheetsRequest processes a request, *t*, for map sheets data from the Macintosh. This implementation relies on an assumption that the sccMapSheetsTrans request will come after the sccTerrainTrans for a terrain patch. The function call is ProcessMapSheetsRequest(t). Table 2.2-23 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCMapSheetsRequest	/simnet/mcc/include/SCC.h
rsp	pointer to SCCMapSheetsResponse	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-23: ProcessMapSheetsRequest Information.

2.2.1.5.3 ProcessExerciseRequest

ProcessExerciseRequest notes the start of an exercise at the Macintosh, specified by *t*. The function call is ProcessExerciseRequest(t). Table 2.2-24 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCExerciseRequest	/simnet/mcc/include/SCC.h
rsp	pointer to SCCExerciseResponse	/simnet/mcc/include/SCC.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
i	int	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
SpawnProcess	Section 2.21.1.26.1.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-24: ProcessExerciseRequest Information.

2.2.1.6 cew.c

/simnet/mcc/SCC/cew.c

cew.c contains routines to handle the Combat Engineering Workstation.

2.2.1.6.1 ProcessCEWInitRequest

ProcessCEWInitRequest processes a request, *t*, which initializes the specified CEW vehicle and loads it into shared memory. The function call is ProcessCEWInitRequest(t). Table 2.2-25 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCCEWInitRequest	/simnet/mcc/include/SCC.h
pars	pointer to CEWParameters	/simnet/mcc/include/MCC_pars.h
organization	OrganizationalUnit	/simnet/common/include/protocol/basic.h
vehicleIndex	short	Standard C type.

Calls	
Function	**Where Described**
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.
GetSCCVehicleIndex	Section 2.2.1.6.4.
GetCEWParameters	Section 2.21.1.8.1.
InitUnit	Section 2.21.1.19.4.
SendConsoleResponse	Section 2.21.1.7.6.
Called By	
Function	**Where Described**
RequestArrived	Section 2.2.1.4.3.

Table 2.2-25: ProcessCEWInitRequest Information.

2.2.1.6.2 ProcessCEWQueryRequest

ProcessCEWQueryRequest returns current information to the Mac CEW console about the specified CEW vehicle. The function call is ProcessCEWQueryRequest (t). Table 2.2-26 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
rsp	pointer to SCCCEWQueryResponse	/simnet/mcc/include/SCC.h
req	pointer to SCCCEWQueryRequest	/simnet/mcc/include/SCC.h
pars	pointer to CEWParameters	/simnet/mcc/include/MCC_pars.h
tmpKind	short	Standard C type.
tmpVehicle	short	Standard C type.
vehicleIndex	int	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
GetSCCVehicleIndex	Section 2.2.1.6.4.	
GetCEWParameters	Section 2.21.1.8.1.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-26: ProcessCEWQueryRequest Information.

2.2.1.6.3 ProcessCEWStartRequest

ProcessCEWStartRequest spawns the CEW process and loads the number of each type of CEW vehicle into shared memory. The function call is ProcessCEWStartRequest(t). Table 2.2-27 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCCEWStartRequest	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SpawnProcess	Section 2.21.1.26.1.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-27: ProcessCEWStartRequest Information.

2.2.1.6.4 GetSCCVehicleIndex

GetSCCVehicleIndex returns the index used by the SCC process given the vehicle identifier, *sccVehicle*, passed in by the SCC console. The function call is GetSCCVehicleIndex(sccVehicle). Table 2.2-28 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sccVehicle	short	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
sccVehicle -1	short	Vehicle index.
Called By		
Function	**Where Described**	
ProcessCEWInitRequest	Section 2.2.1.6.1.	
ProcessCEWQueryRequest	Section 2.2.1.6.2.	

Table 2.2-28: GetSCCVehicleIndex Information.

2.2.1.6.5 ProcessCEWReconstRequest

ProcessCEWReconstRequest redirects the CEW vehicle reconstitute message to the CEW process. The function call is ProcessCEWReconstRequest(t). Table 2.2-29 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCCEWReconstRequest	/simnet/mcc/include/SCC.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-29: ProcessCEWReconstRequest Information.

2.2.2 Placing Combat Vehicles

The SCC is notified when a simulator is to be allocated to a company or placed on the terrain. The SCC then notifies any Placement consoles of this information. It sends the place information to Mother and waits asynchronously for a message from Mother if the simulator failed to be activated. It contains one CSU, sim.c.

2.2.2.1 sim.c

/simnet/mcc/SCC/sim.c

sim.c contains routines which implement the control of the vehicle simulators from the SIMNET Control Console. It also handles some aspects of reconstituting vehicle simulators.

2.2.2.1.1 ProcessSimAllocRequest

ProcessSimAllocRequest processes a notification, *t*, from the SCC Macintosh that a simulator has been allocated to a company. The function call is ProcessSimAllocRequest(t). Table 2.2-30 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SimAllocRequest	/simnet/mcc/include/sim_xact.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
vehicle	unsigned short	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-30: ProcessSimAllocRequest Information.

2.2.2.1.2 SimPlacedLocally

SimPlacedLocally tells the Placement process that the SCC console has been used to place a vehicle, *vehicle*. The function call is SimPlacedLocally(vehicle). Table 2.2-31 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	unsigned short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-31: SimPlacedLocally Information.

2.2.2.1.3 SimAllocedRemotely

SimAllocedRemotely tells the SCC console that a certain simulator has been activated from a Placement console. The function call is SimAllocedRemotely(msg). Table 2.2-32 describes the parameters used and functions called using this function. *msg* is an MCC message buffer that contains the vehicle that was allocated and the company to which it was allocated in the "allocated" structure.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SimAllocRequest	/simnet/mcc/include/sim_xact.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.2.1.4.2.	

Table 2.2-32: SimAllocedRemotely Information.

2.2.2.1.4 SimPlacedRemotely

SimPlacedRemotely tells the SCC console that a certain simulator, specified by *vehicle*, has been activated from a Placement console. The function call is SimPlacedRemotely(vehicle). Table 2.2-33 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	unsigned short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
req	SimPlacedRequest	/simnet/mcc/include/sim_xact.h
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	

Called By	
Function	**Where Described**
ProcessMessage	Section 2.2.1.4.2.

Table 2.2-33: SimPlacedRemotely Information.

2.2.2.1.5 ReportSimProblem

ReportSimProblem reports a problem, *problem*, with the simulator, *vehicle*. The function call is ReportSimProblem(problem, vehicle). Table 2.2-34 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
problem	SimProblem	/simnet/mcc/Include/MCC_ipc .h
vehicle	unsigned short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCSimProblemRequest	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.2.1.4.2.	

Table 2.2-34: ReportSimProblem Information.

2.2.3 Placing Static Vehicles

The SCC is used to initialize the other MCC consoles and their CCVs. There are some CCVs which do not belong to any other console like the TOC and ALOC CCVs. These are simply placed by the SCC. Other CCVs, such as fuel trucks, ammo trucks, mortars or howitzers, are also given an initial location when their console is to be started. The SCC also provides a way to place gunnery targets, which can be any kind of vehicle simulated by the MCC to be placed on the terrain. These gunnery targets do not move across the terrain unless they are reconstituted. Their only movement occurs if they have a turret, in which case their turret slews back and forth continuously. Static vehicle placement is also handled by css.c in Section 2.2.1.2 and fse.c in Section 2.2.1.3. Please reference these sections for additional information. This second level CSC contains the following CSUs:

ccv.c
target.c

2.2.3.1 ccv.c
/simnet/mcc/SCC/ccv.c

ccv.c contains routines which place computer-controlled vehicles, like the TOC components and the Admin/Log Center, on the terrain.

2.2.3.1.1 ProcessTOCRequest

ProcessTOCRequest places the TOC on the terrain. The transaction, *t*, contains data for a SCCTOCRequest. This holds the configuration in which to place the vehicles of the TOC and the central location. It also contains the alignment of the TOC. The function call is ProcessTOCRequest(t). Table 2.2-35 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCTOCRequest	/simnet/mcc/include/SCC.h
s2	LongPt	/simnet/common/include/global/longpt.h
s3	LongPt	/simnet/common/include/global/longpt.h
fs	LongPt	/simnet/common/include/global/longpt.h
ta	LongPt	/simnet/common/include/global/longpt.h
s2A	int	Standard C type.
s3A	int	Standard C type.
fsA	int	Standard C type.
taA	int	Standard C type.
organization	OrganizationalUnit	/simnet/common/include/protocol/basic.h
marking	VehicleMarking	/simnet/common/include/protocol/basic.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
GetGuises	Section 2.21.1.17.1.	
InitUnit	Section 2.21.1.19.4.	
DisplayCCV	Section 2.21.1.6.1.	
TurnRadioOn	Section 2.21.1.23.3.	
SendConsoleResponse	Section 2.21.1.7.6.	

Called By	
Function	**Where Described**
RequestArrived	Section 2.2.1.4.3.

Table 2.2-35: ProcessTOCRequest Information.

2.2.3.1.2 ProcessTOCDispatchRequest

ProcessTOCDispatchRequest removes the TOC from the terrain. The transaction, *t*, tells the SCC to hide the vehicles of the TOC and does not contain any other relevant information. The function call is ProcessTOCDispatchRequest(t). Table 2.2-36 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Calls		
Function	**Where Described**	
SendConsoleResponse	Section 2.21.1.7.6.	
HideCCV	Section 2.21.1.6.2.	
TurnRadioOff	Section 2.21.1.23.1.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-36: ProcessTOCDispatchRequest Information.

2.2.3.1.3 ProcessALOCRequest

ProcessALOCRequest places the Admin/Log Center on the terrain. *t* contains an SCCALOCRequest which defines the alignment of the ALOC and the location. The function call is ProcessALOCRequest(t). Table 2.2-37 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCALOCRequest	/simnet/mcc/include/SCC.h
organization	OrganizationalUnit	/simnet/common/include/protocol/basic.h
marking	VehicleMarking	/simnet/common/include/protocol/basic.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h

Calls	
Function	**Where Described**
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.
GetGuises	Section 2.21.1.17.1.
InitUnit	Section 2.21.1.19.4.
DisplayCCV	Section 2.21.1.6.1.
TurnRadioOn	Section 2.21.1.23.3.
SendConsoleResponse	Section 2.21.1.7.6.
Called By	
Function	**Where Described**
RequestArrived	Section 2.2.1.4.3.

Table 2.2-37: ProcessALOCRequest Information.

2.2.3.1.4 ProcessALOCDispatchRequest

ProcessALOCDispatchRequest removes the Admin/Log Center from the terrain. *t* is an ALOCDispatchRequest, and does not contain any other information. The function call is ProcessALOCDispatchRequest(t). Table 2.2-38 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Calls		
Function	**Where Described**	
SendConsoleResponse	Section 2.21.1.7.6.	
HideCCV	Section 2.21.1.6.2.	
TurnRadioOff	Section 2.21.1.23.4.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-38: ProcessALOCDispatchRequest Information.

2.2.3.2 target.c

/simnet/mcc/SCC/target.c

target.c contains routines which implement the gunnery targets. Table 2.2-39 shows the variables used by target.c

Variables		
Variable	**Type**	**Where Typedef Declared**
vehTypeTable	array of ObjectType	/simnet/common/include/protocol/basic.h

Table 2.2-39: target.c External Variable Information.

2.2.3.2.1 ProcessTargetSetRequest

ProcessTargetSetRequest processes a request, *t*, to place or remove a target. The function call is ProcessTargetSetRequest(t). Table 2.2-40 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to SCCTargetSetRequest	/simnet/mcc/include/SCC.h
marking	VehicleMarking	/simnet/common/include/protocol/basic.h
unit	OrganizationalUnit	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
Errors		
Error Name	**Reason for Error**	
MCC_INCONSISTENCY	Internal inconsistency.	
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
HideCCV	Section 2.21.1.6.2.	
GetGuises	Section 2.21.1.17.1.	
InitUnit	Section 2.21.1.19.4.	
DisplayCCV	Section 2.21.1.6.1.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-40: ProcessTargetSetRequest Information.

2.2.3.2.2 TargetKilled

TargetKilled tells a Macintosh when a gunnery target, specified by *vehicle*, has been destroyed. The function call is TargetKilled(vehicle). Table 2.2-41 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCTargetHitRequest	/simnet/mcc/include/SCC.h
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.2.1.4.2.	

Table 2.2-41: TargetKilled Information.

2.2.4 Communication with World

The SCC listens to messages that come from Mother. These messages are generated when a packet has come in on the network in which it is interested. It also listens to all of the messages coming from the SCC Macintosh, such as placing a simulator, reconstituting a vehicle, starting a console, etc. It keeps track of whether a console has started properly and retries every 30 seconds if it hasn't started yet. When the SCC is sent a message to end the exercise, it passes this message on to each of the spawned processes so that they can recycle themselves. The CSUs are:

restart.c
stop.c

2.2.4.1 restart.c

/simnet/mcc/SCC/restart.c

restart.c contains routines which attempt to restart a console through the following steps:

1. The console process is sent an exist message, causing it to recycle its Macintosh and exit.
2. There is a 10 second delay in order to let things stabilize and the Macintosh reboot.
3. The console process is killed if it still exists.
4. The console process is respawned.

2.2.4.1.1 RestartConsole

RestartConsole restarts a console identified by process number, *p*. The function call is RestartConsole(p). Table 2.2-42 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
p	int	Standard C type.

Calls	
Function	**Where Described**
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h
SetAlarm	Section 2.21.2.4.2.
Called By	
Function	**Where Described**
ProcessMessage	Section 2.2.1.4.2.

Table 2.2-42: RestartConsole Information.

2.2.4.1.2 Respawn

Respawn kills and then restarts a process identified by number. This process is called by the Alarm package. The function call is Respawn(p, a). Table 2.2-43 describes the parameters used, errors returned and functions called using this function. *p* is the process number of the console to be restarted.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
p	long	Standard C type.
a	pointer to Alarm	/simnet/rcs/libipc/alarm.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pid	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_RESPAWN_BEFORE_SPAWN	Process was never originally spawned, which means it was never initialized at the SCC.	
Calls		
Function	**Where Described**	
SpawnProcess	Section 2.21.1.26.1.	

Table 2.2-43: Respawn Information.

2.2.4.2 stop.c

/simnet/mcc/SCC/stop.c

stop.c contains routines for stopping the SIMNET Control Console. Table 2.2-44 shows the variables used by stop.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
errno	extern int	Standard C type.

Table 2.2-44: stop.c External Variables Information.

2.2.4.2.1 ProcessStopRequest

ProcessStopRequest shuts down the exercise. The function call is ProcessStopRequest(t). Table 2.2-45 describes the parameters used and functions called using this function. *t* is the transaction that contains information telling the other processes to stop.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Calls		
Function	**Where Described**	
ATPResponse	Section 2.21.1.1.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.2.1.4.3.	

Table 2.2-45: ProcessStopRequest Information.

2.2.4.2.2 ProcessExitMessage

ProcessExitMessage is called when a technician terminates the MCC exercise. The function call is ProcessExitMessage(). Table 2.2-46 describes the functions called by this function.

Calls	
Function	**Where Described**
ShutdownChildren	Section 2.2.4.2.4.
RemoveRadios	Section 2.21.1.23.2.
ATPTransact	Section 2.21.1.7.3.
CloseSocket	Section 2.21.2.10.4.
Called By	
Function	**Where Described**
ProcessMessage	Section 2.2.1.4.2.

Table 2.2-46: ProcessExitMessage Information.

2.2.4.2.3 ExerciseStopped

ExerciseStopped is called when the BattleMaster terminates the MCC exercise. The function call is ExerciseStopped(t). Table 2.2-47 describes the parameters used, errors returned and functions called using this function. *t* holds no relevant information.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
ShutdownChildren	Section 2.2.4.2.4.	
RemoveRadios	Section 2.21.1.23.2.	
SendMCCMsg0	Macro defined in /simnet/mcc/include/MCC_ipc.h.	

Table 2.2-47: ExerciseStopped Information.

2.2.4.2.4 ShutdownChildren

ShutdownChildren sends an exit message to all children and waits until they exit. The function call is ShutdownChildren(). Table 2.2-48 describes the functions called by this function.

Calls	
Function	**Where Described**
BroadcastToChildren	Section 2.2.1.4.5.
Called By	
Function	**Where Described**
ExerciseStopped	Section 2.2.4.2.3.
ProcessExitMessage	Section 2.2.4.2.2.

Table 2.2-48: ShutdownChildren Information.

2.3 The Place (Placement) Process

The Place process places combat vehicles on the battlefield at the request of the other console processes. Up to three Placement processes are allowed. Each Place process is directly associated with a particular Place console. The code for placing vehicles on the battlefield in the Place process is virtually the same as the code for vehicle placement in the SCC process. Much of this common code is located in libmcc (See Section 2.21). The structure of the Place process is shown in Figure 2.3-1.

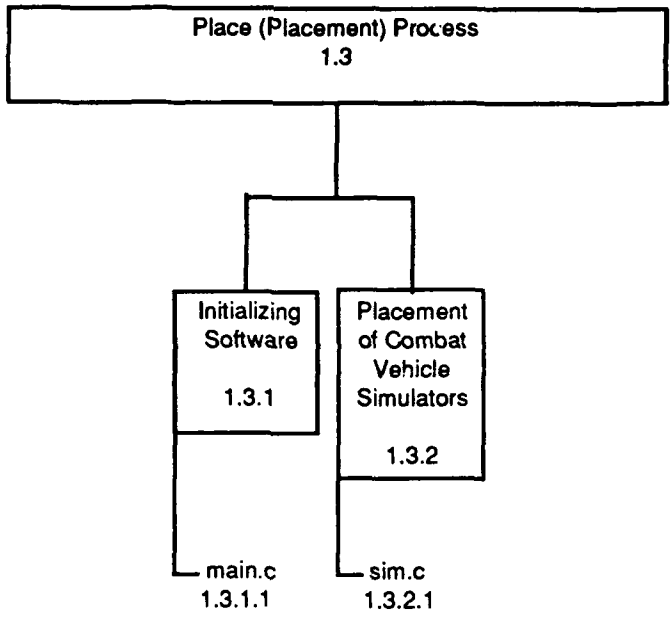


Figure 2.3-3: Place (Placement) Process Structure.

Table 2.3-1 shows the interprocess messages generated and received by the Place process.

Interprocess Message	Received From	Sent To	In Response To	Data Sent	Transaction Type
SimAllocated	SCC		Combat vehicle simulator allocated to company at SCC Console.	- identifier - company	One-way interaction
ActivateVehicle		Mother	Activate or reconstitute combat vehicle simulator	- identifier - reason - type - location - alignment - bumper # - hull and turret orientation - supplies and failure status	First part of two-way interaction
SimPlaced	SCC		Combat vehicle simulator placed at SCC Console	-identifier	One- way interaction
ActivateResult	Mother		Placed combat vehicle .	- identifier - result	Second part of two-way interaction ActivateVehicle

Table 2.3-1: Place Process Interprocess Messages

The SimAllocated message is sent to the Place process by the SCC process if a combat vehicle simulator has been allocated to a company at the SCC console. The Place process can then update all of the Placement consoles accordingly.

The ActivateVehicle is sent to the Mother process by the SCC process or the Place process when it wants to activate or reconstitute a vehicle simulator. This message is the first part of a round-trip interaction whose reply is an ActivateResult message.

The SimPlaced message is sent to the Place process by the SCC process if a combat vehicle simulator has been placed at the SCC console. The Place process can then update all the Placement consoles accordingly.

ActivateResult: This message is sent by the Mother process after it has placed a combat vehicle or CCV on the battlefield. The possible results are:

- success;
- simulator not in expected state;
- no vehicle ID available at this time; or
- no good place to put the vehicle within a radius around the given location.

Second level CSCs associated with the Placement process are:

Initialization Software
Placement of Combat Vehicle Simulators

2.3.1 Initialization Software

The Placement process is spawned by the SCC process when an exercise is started. The Placement process normally sends terrain information to the Placement console application. In the event that the Placement console application has been restarted, any information that the Placement process may have on all other combat simulators will be sent.

The CSU associated with the Initializing Software is main.c.

2.3.1.1 main.c
/simnet/mcc/Place/main.c

main.c contains the top-level code for the Vehicle Placement Console host process.

2.3.1.1.1 main

main is the entry point to the Place process. The function call is main(argc, argv). Table 2.3-2 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Calls		
Function	**Where Described**	
InitializeProcess	Section 2.21.1.19.1.	
OpenHostSocket	Section 2.21.1.7.1.	
ActivateConsole	Section 2.21.1.7.2.	
AlarmsEnabled	Section 2.21.2.4.4.	

Table 2.3-2: main Information.

2.3.1.1.2 ProcessMessage

ProcessMessage processes a message from another process. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the length of the data in the message received. *data* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. The function call is ProcessMessage(type, kind, length, data). Table 2.3-3 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	long	Standard C type.
length	int	Standard C type.
data	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
alarmsEnabled	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESSAGE	Message of unexpected kind.	
Calls		
Function	**Where Described**	
AlarmsEnabled	Section 2.21.2.4.4.	
ATPTransact	Section 2.21.1.7.3.	
CloseSocket	Section 2.21.2.10.4.	
ResetClock	Section 2.3.1.1.5.	
SimAllocedRemotely	Section 2.3.2.1.1.	
SimPlacedRemotely	Section 2.3.2.1.3.	

Table 2.3-3: ProcessMessage Information.

2.3.1.1.3 RequestArrived

RequestArrived processes an ATP Request, *t*, from the Place Macintosh. The function call is RequestArrived(t). Table 2.3-4 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vehicle	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_GETREQUEST_FAILED	ATPGetRequest failed.	
MCC_UNKNOWN_REQUEST	Unlnown request from Macintosh.	

Calls	
Function	**Where Described**
ATPGetRequest	Section 2.21.1.1.5.
ProcessGridInfoRequest	Section 2.21.1.16.1.
ProcessStartRequest	Section 2.3.1.1.4.
ProcessSimExistsRequest	Section 2.21.1.25.1.
ProcessSimQueryRequest	Section 2.21.1.25.4.
ProcessSimInitRequest	Section 2.21.1.25.2.
SimPlacedLocally	Section 2.3.2.1.2.

Table 2.3-4: RequestArrived Information.

2.3.1.1.4 ProcessStartRequest

ProcessStartRequest handles a request, *t*, from the Macintosh for information about the exercise participants. The function call is ProcessStartRequest(t). Table 2.3-5 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
rsp	register pointer to PlaceStartResponse	/simnet/mc/include/Place.h
i	int	Standard C type.
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.3.1.1.3.	

Table 2.3-5: ProcessStartRequest Information.

2.3.1.1.5 ResetClock

ResetClock tells the Macintosh what time it is. The function call is ResetClock(). Table 2.3-6 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	PlaceClockRequest	/simnet/mcc/include/Place.h
i	int	Standard C type.

Calls	
Function	**Where Described**
MacintoshTime	Section 2.21.1.27.1.
ATPPut	Section 2.21.1.7.4.
Called By	
Function	**Where Described**
ProcessMessage	Section 2.3.1.1.2.

Table 2.3-6: ResetClock Information.

2.3.2 Placement of Combat Vehicle Simulators

After a combat vehicle simulator has been allocated to a company at the SCC console, a message is sent to all of the Placement processes indicating that the allocation has taken place. After a combat vehicle simulator has been allocated, it can be placed by a placement console.

The Placement process receives messages from the SCC process when a particular combat vehicle simulator has been placed at the SCC console. The Placement process must send a message to the SCC process when a combat vehicle simulator has been placed from a Place console.

The CSU associated with this second level CSC is sim.c.

2.3.2.1 sim.c
/simnet/mcc/Place/sim.c

sim.c contains routines to implement the control of vehicle simulators from the SIMNET Placement Console.

2.3.2.1.1 SimAllocedRemotely

SimAllocedRemotely processes a notification, *msg*, from the SCC process that a simulator has been allocated to a company. The function call is SimAllocedRemotely(msg). Table 2.3-7 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SimAllocRequest	/simnet/mc/include/sim_xact.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	

Called By	
Function	**Where Described**
ProcessMessage	Section 2.3.1.1.2.

Table 2.3-7: SimAllocedRemotely Information.

2.3.2.1.2 SimPlacedLocally

SimPlacedLocally tells the Placement console host process that the SCC console has been used to place a vehicle, *vehicle*. The function call is SimPlacedLocally(vehicle). Table 2.3-8 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	int	Standard C type.
Internal Var'ables		
Variable	**Type**	**Where Typedef Declared**
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
SimPlacedRemotely	Section 2.3.2.1.3.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.3.1.1.3.	

Table 2.3-8: SimPlacedLocally Information.

2.3.2.1.3 SimPlacedRemotely

SimPlacedRemotely tells the Placement consoles that a certain simulator, specified by *vehicle*, has been activated from another Placement console. The function call is SimPlacedRemotely(vehicle). Table 2.3-9 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
req	SimPlacedRequest	/simnet/mcc/include/sim_xact.h
i	short	Standard C type.

Calls	
Function	**Where Described**
ATPPut	Section 2.21.1.7.4.
Called By	
Function	**Where Described**
ProcessMessage	Section 2.3.1.1.2.
SimPlacedLocally	Section 2.3.2.1.2.

Table 2.3-9: SimPlacedRemotely Information.

2.4 The Admin Process

The Admin process is used to control ammunition and fuel trucks. It coordinates dispatching the trucks, causes the trucks to suffer breakdowns, and transfers supplies to combat vehicles from depots (except for Resupply Offer and Resupply Cancel which are sent out by the Admin process). The structure of Admin process is shown in Figure 2.4-1.

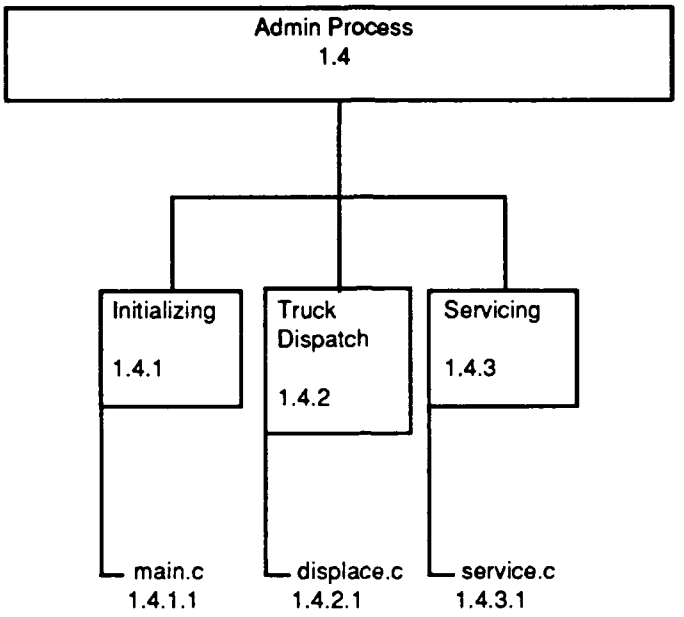


Figure 2.4-1: Admin Process Structure.

It sends out two PDUs, Resupply Offer and Resupply Cancel. It sends and receives the interprocess messages listed in Table 2.4-1.

Interprocess Message	Received From	Sent To	In Response To	Data Sent	Transaction Type
ShowCCV		Mother	CCV needs to be visible	-ownership -identifier -type -alignment -role -company -location -direction	First part of two-way interaction
HideCCV		Mother	CCV needs to be invisible	- indentifier	One-way interaction
KilledCCV	Mother		When CCV killed as shown by these PDUs: *VehicleImpact* Collision IndirectFire	- identifier	One-way interaction
Truck Reconstitute	SCC		Supply truck or maintenance team reconstituted	- identifier - type - location - supplies - company - force	One-way interaction
DepotDisplace	SCC		BattleMaster has displaced ammunition or fuel depots		One-way Interaction
ServiceRequest	Mother		Service Request PDU received from combat vehicle simulator	- truck identifier - combat vehicle identifier	One-way Interaction
Vehicle Resupplied	Mother		Resupply Received PDU from combat vehicle simulator	- truck identifier - combat vehicle identifier - amount of fuel taken - amount of ammo taken	One-way interaction
Resupply Cancel	Mother		Resupply Cancel PDU from combat vehicle simulator	- truck identifier - combat vehicle identifier	One-way interaction

Table 2.4-1: Admin Process Interprocess Messages.

It consists of the following second level CSCs:

Initialization and Communication
Truck Dispatching
Servicing

2.4.1 Initialization and Communication

The Admin Process opens a connection with the Admin Mac and then sends down the initial location of all the ammo and fuel trucks. It listens to messages from the Mother process such as when a truck is killed and passes that information along to the Macintosh. It listens to messages from the Mac indicating when a truck is being dispatched or is transferring supplies, etc. It contains one CSU, main.c.

2.4.1.1 main.c
/simnet/mcc/Admin/main.c

main.c contains the top level code for the Admin/Log Console host process. Table 2.4-2 shows variables used by main.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
m977State[	array numberBnM977Vehicles of TruckState	/simnet/mcc/Admin/Adminhost.h
m978State	array numberBnM978Vehicles of TruckState	/simnet/mcc/Admin/Adminhost.h

Table 2.4-2: main Variable Information.

2.4.1.1.1 main

main is the entry point to the Admin/Log process. The function call is main(argc, argv). Table 2.4-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.
Calls		
Function	**Where Described**	
InitializeProcess	Section 2.21.1.19.1.	
InitializeResupply	Section 2.4.3.1.1.	
OpenHostSocket	Section 2.21.1.7.1.	
ActivateConsole	Section 2.21.1.7.2.	
AlarmsEnabled	Section 2.21.2.4.4.	

Table 2.4-3: main Information.

2.4.1.1.2 ProcessMessage

ProcessMessage processes a message from another process. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the length of the data in the message received. *data* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. The function call is ProcessMessage(type, kind, length, data). Table 2.4-4 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	long	Standard C type.
length	int	Standard C type.
data	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
alarmsEnabled	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESSAGE	Message of unexpected kind.	
Calls		
Function	**Where Described**	
AlarmsEnabled	Section 2.21.2.4.4.	
ATPTransact	Section 2.21.1.7.3.	
CloseSocket	Section 2.21.2.10.4.	
ResetClock	Section 2.4.1.1.3.	
TruckKilled	Section 2.4.2.1.5.	
SendDepotRequest	Section 2.4.1.1.6.	
ServiceRequest	Section 2.4.3.1.2.	
VehicleResupplied	Section 2.4.3.1.3.	
CancelResupply	Section 2.4.3.1.4.	
ReconstituteTruck	Section 2.4.2.1.2.	

Table 2.4-4: ProcessMessage Information.

2.4.1.1.3 ResetClock

ResetClock updates the clock on the Macintosh. The function call is ResetClock(). Table 2.4-5 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	AdminLogClockRequest	/simnet/mcc/include/Admin.h

Calls	
Function	**Where Described**
MacintoshTime	Section 2.21.1.27.1.
ATPPut	Section 2.21.1.7.4.
Called By	
Function	**Where Described**
ProcessMessage	Section 2.4.1.1.2.

Table 2.4-5: ResetClock Information.

2.4.1.1.4 RequestArrived

RequestArrived processes an ATP Request, *t*, from the Admin/Log Macintosh. The function call is RequestArrived(t). Table 2.4-6 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Errors		
Error Name	**Reason for Error**	
MCC_GETREQUEST_FAILED	ATPGetRequest failed.	
MCC_UNKNOWN_REQUEST	Unknown request from Macintosh.	
Calls		
Function	**Where Described**	
ATPGetRequest	Section 2.21.1.1.5.	
ProcessGridInfoRequest	Section 2.21.1.16.1.	
SendForceRequest	Section 2.4.1.1.5.	
InitializeAllTrucks	Section 2.4.1.1.7.	
TruckDispatched	Section 2.4.2.1.4.	
TruckArrived	Section 2.4.2.1.3.	

Table 2.4-6: RequestArrived Information.

2.4.1.1.5 SendForceRequest

SendForceRequest downloads the battalion and company ForceIDs to the Macintosh. The function call is SendForceRequest(). Table 2.4-7 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	AdminLogForceRequest	/simnet/mcc/include/Admin.h
i	int	Standard C type.
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	

Called By	
Function	**Where Described**
RequestArrived	Section 2.4.1.1.4.

Table 2.4-7: SendForceRequest Information.

2.4.1.1.6 SendDepotRequest

SendDepotRequest downloads the depot locations to the Macintosh. The function call is SendDepotRequest(). Table 2.4-8 describes the internal variables used, errors returned and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	AdminLogDepotRequest	/simnet/mcc/include/Admin.h
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
InitializeAllTrucks	Section 2.4.1.1.7.	
ProcessMessage	Section 2.4.1.1.2.	

Table 2.4-8: SendDepotRequest Information.

2.4.1.1.7 InitializeAllTrucks

InitializeAllTrucks ships initial parameters to the Macintosh. The function call is InitializeAllTrucks(). Table 2.4-9 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
i	register unsigned short	Standard C type.
sup	AdminSuppliesRequest	/simnet/mcc/include/Admin.h
Calls		
Function	**Where Described**	
InitializeTruck	Section 2.4.2.1.1.	
SendDepotRequest	Section 2.4.1.1.6.	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.4.1.1.4.	

Table 2.4-9: InitializeAllTrucks Information.

2.4.1.1.8 LookupTruckNumber

LookupTruckNumber finds a truck state record given *truckNumber* and *capabilities*. The function call is LookupTruckNumber(truckNumber, capabilities). Table 2.4-10 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truckNumber	int	Standard C type.
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
Return Values		
Return Value	**Type**	**Meaning**
&m977State[truckNumber]	pointer to TruckState	data which describes the ammo truck indexed by truckNumber
&m978State[truckNumber]	pointer to TruckState	data which describes the fuel truck indexed by truckNumber
Called By		
Function	**Where Described**	
ReconstituteTruck	Section 2.4.2.1.2.	
TruckArrived	Section 2.4.2.1.3.	
TruckDispatched	Section 2.4.2.1.4.	

Table 2.4-10: LookupTruckNumber Information.

2.4.1.1.9 LookupCCVNumber

LookupCCVNumber finds a CCV state record, specified by *ccvNumber*. Given the CCV number of the vehicle, return the data which corresponds to it. CCV numbers are unique among all MCC vehicles. The function call is LookupCCVNumber(ccvNumber). Table 2.4-11 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
ccvNumber	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
&m977State[ccvNumber - firstBnM977Vehicle]	int	the CCV state record
&m978State[ccvNumber - firstBnM978Vehicle]	int	the CCV state record

Called By	
Function	**Where Described**
TruckKilled	Section 2.4.2.1.5.
ServiceRequest	Section 2.4.3.1.2.
VehicleResupplied	Section 2.4.3.1.3.
CancelResupply	Section 2.4.3.1.4.

Table 2.4-11: LookupCCVNumber Information.

2.4.2 Truck Dispatching

When Admin receives a message from the Mac indicating a truck is to be moved, Admin sends a message to Mother to deactivate the truck. The Admin Mac sends another message when it's time for a truck to arrive at its destination. The Admin Process sends a message to Mother to place the CCV again at its new location. The only CSU is displace.c.

2.4.2.1 displace.c
/simnet/mcc/Admin/displace.c

displace.c contains routines for placing and displacing trucks controlled from the Admin/Log Console.

2.4.2.1.1 InitializeTruck

InitializeTruck downloads the initial parameters for a single truck to the Macintosh. *state* points to a local structure which contains more information on the truck; *pars* is the data structure in shared memory which holds some information on the truck. The function call is InitializeTruck(state, pars). Table 2.4-12 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	AdminLogTruckRequest	/simnet/mcc/include/Admin.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
RestoreTruck	Section 2.4.2.1.6.	
GetGuises	Section 2.21.1.17.1.	
DisplayCCV	Section 2.21.1.6.1.	
ATPTransact	Section 2.21.1.7.3.	

Called By	
Function	**Where Described**
ReconstituteTruck	Section 2.4.2.1.2.
InitializeAllTrucks	Section 2.4.1.1.7.

Table 2.4-12: InitializeTruck Information.

2.4.2.1.2 ReconstituteTruck

ReconstituteTruck is called to reset a truck to the ready state with new position, load and company assignments. *msg* contains the truck's vehicle number, its capabilities, a location, and the organization. The function call is ReconstituteTruck(msg). Table 2.4-13 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
pars	register pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
Calls		
Function	**Where Described**	
LookupTruckNumber	Section 2.4.1.1.8.	
GetTruckParameters	Section 2.21.1.29.1.	
Lock	Section 2.21.2.6.3.	
Unlock	Section 2.21.2.6.4.	
InitializeTruck	Section 2.4.2.1.1.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.4.1.1.2.	

Table 2.4-13: ReconstituteTruck Information.

2.4.2.1.3 TruckArrived

TruckArrived is called when a truck arrives at its destination. *t* contains an AdminLogArriveRequest which holds information about the truck such as its fleet, its vehicle number, and its load of ammunition. The function call is TruckArrived(t). Table 2.4-14 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
req	pointer to AdminLogArriveRequest	/simnet/mcc/include/Admin.h
rsp	pointer to AdminLogArriveResponse	/simnet/mcc/include/Admin.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
TurnFleetIntoCapabilities	Section 2.4.2.1.7.	
LookupTruckNumber	Section 2.4.1.1.8.	
GetTruckParameters	Section 2.21.1.29.1.	
Lock	Section 2.21.2.6.3.	
Unlock	Section 2.21.2.6.4.	
RestoreTruck	Section 2.4.2.1.6.	
GetGuises	Section 2.21.1.17.1.	
DisplayCCV	Section 2.21.1.6.1.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.4.1.1.4.	

Table 2.4-14: TruckArrived Information.

2.4.2.1.4 TruckDispatched

TruckDispatched is called when a truck has been dispatched. *t* contains fleet and vehicle number of the truck to dispatch. The function call is TruckDispatched(t). Table 2.4-15 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to AdminLogDispatchRequest	/simnet/mcc/include/Admin.h
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h

Calls	
Function	**Where Described**
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.
TurnFleetIntoCapabilities	Section 2.4.2.1.7.
LookupTruckNumber	Section 2.4.1.1.8.
SendConsoleResponse	Section 2.21.1.7.6.
RestoreTruck	Section 2.4.2.1.6.
HideCCV	Section 2.21.1.6.2.
Called By	
Function	**Where Described**
RequestArrived	Section 2.4.1.1.4.

Table 2.4-15: TruckDispatched Information.

2.4.2.1.5 TruckKilled

TruckKilled is called when a truck has been destroyed. *msg* is the CCV number of the truck killed. The function call is TruckKilled(msg). Table 2.4-16 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
req	AdminLogKilledRequest	/simnet/mcc/include/Admin.h
Calls		
Function	**Where Described**	
LookupCCVNumber	Section 2.4.1.1.8.	
RestoreTruck	Section 2.4.2.1.6.	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.4.1.1.2.	

Table 2.4-16: TruckKilled Information.

2.4.2.1.6 RestoreTruck

RestoreTruck is called when a truck is stationary, visible, healthy and alone. The function call is RestoreTruck(state). Table 2.4-17 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
Calls		
Function	**Where Described**	
CancelAlarm	Section 2.21.2.4.3.	
CancelOffers	Section 2.4.3.1.6.	
Called By		
Function	**Where Described**	
TruckDispatched	Section 2.4.2.1.4.	
TruckKilled	Section 2.4.2.1.5.	
InitializeTruck	Section 2.4.2.1.1.	
TruckArrived	Section 2.4.2.1.3.	

Table 2.4-17: RestoreTruck Information.

2.4.2.1.7 TurnFleetIntoCapabilities

TurnFleetIntoCapabilities fills in the proper fields of a capabilities structure from a fleet type, where *fleet* is ammunition, fuel, pallet, or maintenance; *cap* is a vehicle capabilities structure that marks TRUE with all the capabilities of the truck. The function call is TurnFleetIntoCapabilities(fleet, cap). Table 2.4-18 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
fleet	int	Standard C type.
cap	pointer to VehicleCapabilities	/simnet/common/include/protocol/basic.h
Called By		
Function	**Where Described**	
TruckArrived	Section 2.4.2.1.3.	
TruckDispatched	Section 2.4.2.1.4.	

Table 2.4-18: TurnFleetIntoCapabilities Information.

2.4.3 Servicing

Admin receives messages from Mother when a combat vehicle is close enough to a truck and needs to be resupplied with fuel or ammo. Admin changes the internal state of the truck to servicing and sends out packets which offer fuel or ammo. The combat vehicle sends messages via Mother describing how much fuel or ammo was taken. Admin then propagates the information to the Admin Mac so it can decrement the truck load. This functionality is realized by the CSU service.c.

2.4.3.1 service.c
/simnet/mcc/Admin/service.c

service.c contains routines which implement the servicing of the vehicles by the Admin/Log trucks. Table 2.4-19 describes the variables used by service.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
roSimPDU	staticSimulationPDU	/simnet/common/include/protocol/p_sim.h
rcSimPDU	staticSimulationPDU	/simnet/common/include/protocol/p_sim.h
rofferPDU	pointer to ResupplyVariant	/simnet/common/include/protocol/p_sim.h
rcancelPDU	pointer to ResupplyVariant	/simnet/common/include/protocol/p_sim.h
newAmmoType	array of ObjectType	/simnet/common/include/protocol/basic.h
networkInterface	extern int	Standard C type.

Table 2.4-19: service.c Variable Information.

2.4.3.1.1 InitializeResupply

InitializeResupply initializes the resupply package. The function call is InitializeResupply(). Table 2.4-20 describes this function.

Called By	
Function	**Where Described**
main	Section 2.4.1.1.1.

Table 2.4-20: InitializeResupply Information.

2.4.3.1.2 ServiceRequest

ServiceRequest is called when a combat vehicle wants service from a truck, as indicated by *msg*. The function call is ServiceRequest(msg). Table 2.4-21 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
i	register unsigned short	Standard C type.
client	register pointer to struct client	/simnet/mcc/Admin/Adminhost.h
mun	pointer to MunitionQuantity	/simnet/common/include/protocol/basic.h
numMun	unsigned char	Standard C type.
t	short	Standard C type.
quantum	short	Standard C type.
req	AdminLogServicingRequest	/simnet/mcc/include/Admin.h
length	long	Standard C type.
msg1	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
LookupCCVNumber	Section 2.4.1.1.9.	
GetTruckParameters	Section 2.21.1.29.1.	
CancelAlarm	Section 2.21.2.4.3.	
ATPPut	Section 2.21.1.7.4	
SetAlarm	Section 2.21.2.4.2.	
PRO_SIM_RESUPPLY_OFFER_SIZE	p_size.h	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
AssocSendDatagram	Section 2.20.1.2.1.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.4.1.1.2.	

Table 2.4-21: ServiceRequest Information.

2.4.3.1.3 VehicleResupplied

VehicleResupplied is called when a vehicle has accepted supplies that have been offered, as indicated by *msg*. The function call is VehicleResupplied(msg). Table 2.4-22 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
i	register unsigned short	Standard C type.
client	register pointer to struct client	/simnet/mcc/Admin/Adminhost.h
req	AdminLogTransferRequest	/simnet/mcc/include/Admin.h
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_PACKET	Unexpected packet type. There is no truck expecting a resupply from the requesting vehicle.	
MCC_TOO_MUCH_AMMO_TAKEN	Took more ammo than offered.	
MCC_TOO_MUCH_FUEL_TAKEN	Took more fuel than offered.	
Calls		
Function	**Where Described**	
LookupCCVNumber	Section 2.4.1.1.9.	
GetTruckParameters	Section 2.21.1.29.1.	
CancelAlarm	Section 2.21.2.4.3.	
RemoveClient	Section 2.4.3.1.7.	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.4.1.1.2.	

Table 2.4-22: VehicleResupplied Information.

2.4.3.1.4 CancelResupply

CancelResupply is called when a Resupply Cancel PDU is received from a client which indicates that a resupply process is to halt. *msg* holds a resupply structure which contains the id of the client requesting the supplies. The function call is CancelResupply(msg). Table 2.4-23 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
i	register unsigned short	Standard C type.
client	register pointer to struct client	/simnet/mcc/Admin/Adminhost.h
req	AdminLogTransferRequest	/simnet/mcc/include/Admin.h
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_PACKET	Unexpected packet type. There was no truck that thought it was in a resupply transaction with this client.	
Calls		
Function	**Where Described**	
LookupCCVNumber	Section 2.4.1.1.9.	
GetTruckParameters	Section 2.21.1.29.1.	
CancelAlarm	Section 2.21.2.4.3.	
WithdrawOffer	Section 2.4.3.1.5.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.4.1.1.2.	

Table 2.4-23: CancelResupply Information.

2.4.3.1.5 WithdrawOffer

WithdrawOffer cancels an offer to a client. *state* is the structure describing the truck that was providing the supply; *client* is the structure describing the vehicle receiving the supply. The function call is WithdrawOffer(state, client). Table 2.4-24 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Adminhost.h
client	register pointer to struct client	/simnet/mcc/Admin/Adminhost.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register unsigned short	Standard C type.
Calls		
Function	**Where Described**	
RemoveClient	Section 2.4.3.1.7.	

Called By	
Function	**Where Described**
ServiceTimeout	Section 2.4.3.1.8.
CancelResupply	Section 2.4.3.1.4.

Table 2.4-24: WithdrawOffer Information.

2.4.3.1.6 CancelOffers

CancelOffers cancels all outstanding offers of supplies. *state* is the structure describing the truck that was providing the supply. The function call is CancelOffers(state). Table 2.4-25 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Admin host.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
client	register pointer to struct client	/simnet/mcc/Admin/Admin host.h
Calls		
Function	**Where Described**	
AssocSendDatagram	Section 2.20.1.2.1.	
PRO_SIM_RESUPPLY_CANCEL_SIZE	p_size.h	
CancelAlarm	Section 2.21.2.4.3.	
Called By		
Function	**Where Described**	
RestoreTruck	Section 2.4.2.1.6.	

Table 2.4-25: CancelOffers Information.

2.4.3.1.7 RemoveClient

RemoveClient removes a client from the list of clients maintained for a particular service truck. *state* is the structure describing the truck that was providing the supply; *client* is the structure describing the vehicle receiving the supply. The function call is RemoveClient(state, client). Table 2.4-26 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Admin/Admin host.h
client	register pointer to struct client	/simnet/mcc/Admin/Admin host.h

Calls	
Function	**Where Described**
SetAlarm	Section 2.21.2.4.2.
Called By	
Function	**Where Described**
WithdrawOffer	Section 2.4.3.1.5.
VehicleResupplied	Section 2.4.3.1.3.

Table 2.4-26: RemoveClient Information.

2.4.3.1.8 ServiceTimeout

ServiceTimeout creates a one minute timeout on an outstanding Resupply Offer PDU. *param* contains the TruckState of the turck offering the supplies; *a* is the alarm which initiated this call. The function call is ServiceTimeout(param, a). Table 2.4-27 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
param	long	Standard C type.
a	Alarm	/simnet/rcs/libipc/alarm.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	pointer to TruckState	/simnet/mcc/Admin/Admin host.h
i	register unsigned short	Standard C type.
client	register pointer to struct client	/simnet/mcc/Admin/Admin host.h
Errors		
Error Name	**Reason for Error**	
MCC_INCONSISTENCY	Internal inconsistency.	
Calls		
Function	**Where Described**	
AssocSendDatagram	Section 2.20.1.2.1.	
PRO_SIM_RESUPPLY_CANCEL_SIZE	p_size.h	
WithdrawOffer	Section 2.4.3.1.5.	

Table 2.4-27: ServiceTimeout Information.

2.4.3.1.9 ReadyTimeout

ReadyTimeout returns a truck to ready state when no additional ServiceRequest PDUs seem to be forthcoming. *param* contains the TruckState of the truck offering the supplies; *a* is the alarm which initiated this call. The function call is ReadyTimeout(param, a). Table 2.4-28 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
param	long	Standard C type.
a	Alarm	/simnet/rcs/libipc/alarm.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	pointer to TruckState	/simnet/mcc/Admin/Admin host.h
req	AdminLogServicingRequest	/simnet/mcc/include/Admin.h
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	

Table 2.4-28: ReadyTimeout Information.

2.5 The Maint (Maintenance) Process

The Maint process is used to control a fleet of maintenance vehicles. All of the maintenance vehicles are currently represented by 2 1/2 ton trucks, even though they occasionally are used to recover vehicles. The maintenance vehicles can perform repairs on up to three vehicles at one time and can be moved and destroyed like other CCVs. The structure of Maint process is shown in Figure 2.5-1.

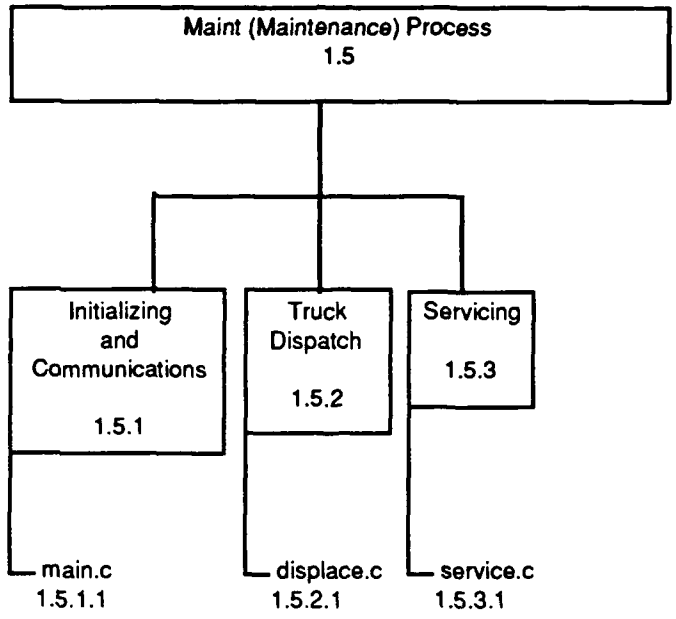


Figure 2.5-1: Maint (Maintenance) Process Structure.

The Maint Process CSC sends and receives the interprocess messages listed in Table 2.5-1.

Interprocess Message	Received From	Sent To	In Response To	Data Sent	Transaction Type
ShowCCV		Mother	CCV needs to be visible	-ownership -identifier -type -alignment -role -company -location -direction	First part of two-way interaction
HideCCV		Mother	CCV needs to be invisible	- indentifier	One-way interaction
KilledCCV	Mother		When CCV killed as shown by these PDUs: VehicleImpact Collision IndirectFire	- identifier	One-way interaction
ActivateVehicle		Mother	Reactivate vehicle when at towing destination	- identifier - reason - type - location - alignment - bumper # - hull and turret orientation - supplies and failure status	First part of two-way interaction
ActivateResult	Mother		Placed combat vehicle or CCV.	- identifier - result	Second part of two-way interaction ShowCCV
Deactivate Vehicle		Mother	Combat vehicle simulator to be towed	- identifier	One-way interaction
Truck Reconstitute	SCC		Supply truck or maintenance team reconstituted	- identifier - type - location - supplies - company - force	One-way interaction
DepotDisplace	SCC		BattleMaster has displaced ammunition or fuel depots		One-way Interaction
ServiceRequest	Mother		Service Request PDU received from combat vehicle simulator	- truck identifier - combat vehicle identifier	One-way Interaction

Table 2.5-1: Maint Process Interprocess Messages.

The Maint Process CSC consists of the following second level CSCs:

Initialization and Communication
Truck Dispatching
Servicing

2.5.1 Initialization and Communication

The Maint Process opens a connection with the Maint Mac and then sends down the initial location of all the maintenance trucks. It listens to messages from the Mother process (such as when a truck is killed) and passes that information along to the Macintosh. It listens to messages from the Mac indicating when a truck is being dispatched or is repairing a vehicle, reconstituted, etc. It contains one CSU, main.c.

2.5.1.1 main.c
/simnet/mcc/Maint/main.c

main.c contains the top level code of the Maintenance Console host process. Table 2.5-2 describes the variables used by main.c.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
maintState	TruckState	/simnet/mcc/Maint/Mainthost.h

Table 2.5-2: main Variable Information.

2.5.1.1.1 main

main is the entry point to the Maint process. The function call is main(argc, argv). Table 2.5-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.
Calls		
Function	**Where Described**	
InitializeProcess	Section 2.21.1.19.1.	
OpenHostSocket	Section 2.21.1.7.1.	
ActivateConsole	Section 2.21.1.7.2.	
AlarmsEnabled	Section 2.21.2.4.4.	

Table 2.5-3: main Information.

2.5.1.1.2 ProcessMessage

ProcessMessage processes a message from another process. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the

length of the data in the message received. *data* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. The function call is ProcessMessage(type, kind, length, data). Table 2.5-4 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	long	Standard C type.
length	int	Standard C type.
data	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
alarmsEnabled	int	Standard C type.
i	int	Standard C type.
req	MaintTowFailedRequest	/simnet/mcc/include/Maint.h
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESSAGE	Message of unexpected kind.	
Calls		
Function	**Where Described**	
AlarmsEnabled	Section 2.21.2.4.4.	
ATPTransact	Section 2.21.1.7.3.	
CloseSocket	Section 2.21.2.10.4.	
ResetClock	Section 2.5.1.1.3.	
TruckKilled	Section 2.5.2.1.5.	
SendDepotRequest	Section 2.5.1.1.6.	
ServiceRequest	Section 2.5.3.1.1.	
ReconstituteTruck	Section 2.5.2.1.2.	

Table 2.5-4: ProcessMessage Information.

2.5.1.1.3 ResetClock

ResetClock updates the clock on the Macintosh. The function call is ResetClock (). Table 2.5-5 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	MaintClockRequest	/simnet/mcc/include/Maint.h
Calls		
Function	**Where Described**	
MacintoshTime	Section 2.21.1.27.1.	
ATPPut	Section 2.21.1.7.4.	

Called By	
Function	**Where Described**
ProcessMessage	Section 2.5.1.1.2.

Table 2.5-5: ResetClock Information.

2.5.1.1.4 RequestArrived

RequestArrived processes an ATP Request, *t*, from the Maint Macintosh. The function call is RequestArrived(t). Table 2.5-6 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Errors		
Error Name	**Reason for Error**	
MCC_GETREQUEST_FAILED	ATPGetRequest failed.	
MCC_UNKNOWN_REQUEST	Unknown request from Macintosh.	
Calls		
Function	**Where Described**	
ATPGetRequest	Section 2.21.1.1.5.	
ProcessGridInfoRequest	Section 2.21.1.16.1.	
SendForceRequest	Section 2.5.1.1.5.	
InitializeAllTrucks	Section 2.5.1.1.7.	
TruckDispatched	Section 2.5.2.1.4.	
TruckArrived	Section 2.5.2.1.3.	
PerformRepair	Section 2.5.3.1.2.	

Table 2.5-6: RequestArrived Information.

2.5.1.1.5 SendForceRequest

SendForceRequest downloads battalion and company ForceIDs to the Macintosh. The function call is SendForceRequest(). Table 2.5-7 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	MaintForceRequest	/simnet/mcc/include/Maint.h
i	int	Standard C type.
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	

Called By	
Function	**Where Described**
RequestArrived	Section 2.5.1.1.4.

Table 2.5-7: SendForceRequest Information.

2.5.1.1.6 SendDepotRequest

SendDepotRequest downloads a depot location to the Macintosh. The function call is SendDepotRequest(). Table 2.5-8 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	MaintDepotRequest	/simnet/mcc/include/Maint.h
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
InitializeAllTrucks	Section 2.5.1.1.7.	
ProcessMessage	Section 2.5.1.1.2.	

Table 2.5-8: SendDepotRequest Information.

2.5.1.1.7 InitializeAllTrucks

InitializeAllTrucks ships the initial truck parameters to the Macintosh. The function call is InitializeAllTrucks(). Table 2.5-9 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
i	register short	Standard C type.
Calls		
Function	**Where Described**	
TurnFleetIntoCapabilities	Section 2.5.2.1.7.	
InitializeTruck	Section 2.5.2.1.1.	
SendDepotRequest	Section 2.5.1.1.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.5.1.1.3.	

Table 2.5-9: InitializeAllTrucks Information.

2.5.1.1.8 LookupTruckNumber

LookupTruckNumber finds a truck state record for a truck specified by *truckNumber* and *capabilities*. The function call is LookupTruckNumber(truckNumber, capabilities). Table 2.5-10 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truckNumber	int	Standard C type.
capabilities	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
&maintState[truckNumber]	int	The truck maintenance state record
Called By		
Function	**Where Described**	
ReconstituteTruck	Section 2.5.2.1.2.	
TruckArrived	Section 2.5.2.1.3.	
TruckDispatched	Section 2.5.2.1.4.	
PerformRepair	Section 2.5.3.1.2.	

Table 2.5-10: LookupTruckNumber Information.

2.5.1.1.9 LookupCCVNumber

LookupCCVNumber finds a CCV state record, specified by *ccvNumber*. Given the CCV number of the vehicle, return the data which corresponds to it. CCV numbers are unique among all MCC vehicles. The function call is LookupCCVNumber(ccvNumber). Table 2.5-11 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
ccvNumber	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
&maintState[ccvNumber - first BnRepairVehicle]	int	The CCV maintenance state record
Called By		
Function	**Where Described**	
TruckKilled	Section 2.5.2.1.5.	
ServiceRequest	Section 2.5.3.1.1.	

Table 2.5-11: LookupCCVNumber Information.

2.5.2 Truck Dispatching

When Maint receives a message from the Mac indicating a truck is to be moved, Maint sends a message to Mother to deactivate the truck. The Maint Mac sends another message when it is time for a truck to arrive at its destination. The Maint Process sends a message to Mother to place the maintenance vehicle again at its new location. The only CSU is displace.c.

2.5.2.1 displace.c
/simnet/mcc/Maint/displace.c

displace.c contains routines for placing and displacing trucks controlled fron the Maintenance Console.

2.5.2.1.1 InitializeTruck

InitializeTruck downloads the initial parameters for a single truck to the Macintosh. *state* points to a local structure which contains more information on the truck; *pars* is the data structure in shared memory which holds some information on the truck. The function call is InitializeTruck(state, pars). Table 2.5-12 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
pars	register pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	MaintTeamRequest	/simnet/mcc/include/Maint.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
RestoreTruck	Section 2.5.2.1.6.	
GetGuises	Section 2.21.1.17.1.	
DisplayCCV	Section 2.21.1.6.1.	
ATPTransact	Section 2.21.1.7.3.	
Called By		
Function	**Where Described**	
InitializeAllTrucks	Section 2.5.1.1.7.	
ReconstituteTruck	Section 2.5.2.1.2.	

Table 2.5-12: InitializeTruck Information.

2.5.2.1.2 ReconstituteTruck

ReconstituteTruck resets a truck to the ready state with a new position, company and alignment. *msg* contains the truck's vehicle number, its capabilities, a location, and the organization. The function call is ReconstituteTruck(msg). Table 2.5-13 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
pars	register pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
Calls		
Function	**Where Described**	
LookupTruckNumber	Section 2.5.1.1.8.	
GetTruckParameters	Section 2.21.1.29.1.	
Lock	Section 2.21.2.6.3.	
Unlock	Section 2.21.2.6.4.	
InitializeTruck	Section 2.5.2.1.1.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.5.1.1.2.	

Table 2.5-13: ReconstituteTruck Information.

2.5.2.1.3 TruckArrived

TruckArrived is called when a truck arrives at its destination. *t* contains an MaintArriveRequest which holds information about the truck, such as its fleet and its vehicle number. The function call is TruckArrived(t). Table 2.5-14 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
req	pointer to MaintArriveRequest	/simnet/mcc/include/Maint.h
rsp	pointer to MaintArriveResponse	/simnet/mcc/include/Maint.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
TurnFleetIntoCapabilities	Section 2.5.2.1.7.	
LookupTruckNumber	Section 2.5.1.1.8.	
GetTruckParameters	Section 2.21.1.29.1.	
RestoreTruck	Section 2.5.2.1.6.	
GetGuises	Section 2.21.1.17.1.	
DisplayCCV	Section 2.21.1.6.1.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.5.1.1.4.	

Table 2.5-14: TruckArrived Information.

2.5.2.1.4 TruckDispatched

TruckDispatched is called when a truck has been dispatched. *t* contains the fleet and vehicle number of the truck to dispatch. The function call is TruckDispatched(t). Table 2.5-15 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to MaintDispatchRequest	/simnet/mcc/include/Maint.h
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
towing	unsigned char	Standard C type.
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
capabilities	VehicleCapabilities	basic.h
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h

Calls	
Function	**Where Described**
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.
TurnFleetIntoCapabilities	Section 2.5.2.1.7.
LookupTruckNumber	Section 2.5.1.1.8.
SendConsoleResponse	Section 2.21.1.7.6.
FQueue_Retrieve	Section 2.21.1.13.7.
RestoreTruck	Section 2.5.2.1.6.
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.
HideCCV	Section 2.21.1.6.2.
Called By	
Function	**Where Described**
RequestArrived	Section 2.5.1.1.4.

Table 2.5-15: TruckDispatched Information.

2.5.2.1.5 TruckKilled

TruckKilled is called when a truck has been destroyed. *msg* is the CCV number of the destroyed truck. The function call is TruckKilled(msg). Table 2.5-16 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
req	MaintKilledRequest	/simnet/mcc/include/Maint.h
Calls		
Function	**Where Described**	
LookupCCVNumber	Section 2.5.1.1.9.	
RestoreTruck	Section 2.5.2.1.6.	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.5.1.1.2.	

Table 2.5-16: TruckKilled Information.

2.5.2.1.6 RestoreTruck

RestoreTruck is called when a truck is now stationary, visible and healthy. The function call is RestoreTruck(state). Table 2.5-17 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pars	pointer to TruckParameters	/simnet/mcc/include/MCC_pars.h
rspKind	long	Standard C type.
rspLen	int	Standard C type.
errCode	int	Standard C type.
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESSAGE	Message of unexpected kind.	
Calls		
Function	**Where Described**	
CancelAlarm	Section 2.21.2.4.3.	
TurnFleetIntoCapabilities	Section 2.5.2.1.7.	
GetTruckParameters	Section 2.21.1.29.1.	
Transact	Section 2.21.2.10.9.	
TraceMessage	Section 2.21.1.19.3.	
Called By		
Function	**Where Described**	
InitializeTruck	Section 2.5.2.1.1.	
TruckArrived	Section 2.5.2.1.3.	
TruckDispatched	Section 2.5.2.1.4.	
TruckKilled	Section 2.5.2.1.5.	

Table 2.5-17: RestoreTruck Information.

2.5.2.1.7 TurnFleetIntoCapabilities

TurnFleetIntoCapabilities takes a fleet and initializes the appropriate fields of a capabilities structure from a fleet type, where *fleet* is either ammunition, fuel, pallet, or maintenance; *cap* is a vehicle capabilities structure that marks TRUE with all the capabilities of the truck. The function call is TurnFleetIntoCapabilities(fleet, cap). Table 2.5-18 describes the parameters used by using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
fleet	int	Standard C type.
cap	pointer to VehicleCapabilities	/simnet/common/include/protocol/basic.h

Called By	
Function	**Where Described**
InitializeAllTrucks	Section 2.5.1.1.7.
TruckArrived	Section 2.5.2.1.3.
TruckDispatched	Section 2.5.2.1.4.
RestoreTruck	Section 2.5.2.1.6.
PerformRepair	Section 2.5.3.1.2.

Table 2.5-18: TurnFleetIntoCapabilities Information.

2.5.3 Servicing

Maint receives messages from Mother when a combat vehicle is close enough to a truck and needs to be repaired or recovered. The operator at the Maint Macintosh selects a repair or designates the vehicle to be recovered. If a repair is to occur, the internal state of the truck is set to servicing and it sends out packets which offer servicing. At the end of the time it takes to perform the selected repair, the Maint process sends out a packet stating that the repair has been completed. If the vehicle moves away from the maintenance team during the repair, the repair is cancelled. If the vehicle is to be recovered, Maint sends a message to Mother to deactivate the CCV and the combat vehicle. When it is time for the team to arrive at its destination, Maint sends Mother another message to place the CCV and reconstitute the combat vehicle at the new location. This functionality is realized by the CSU service.c.

2.5.3.1 service.c
/simnet/mcc/Maint/service.c

service.c contains routines which implement the servicing of vehicles by the maintenance teams. Table 2.5-19 shows variables used by service.c.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
nextRepairEventID	unsigned short	Standard C type.

Table 2.5-19: service.c Variable Information.

2.5.3.1.1 ServiceRequest

ServiceRequest is called when a combat vehicle wants service from a truck, as indicated by *msg*. The function call is ServiceRequest(msg). Table 2.5-20 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
vehicleID	VehicleID	/simnet/common/include/protocol/basic.h
req	MaintServicingRequest	/simnet/mcc/include/Maint.h
veh	pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
Calls		
Function	**Where Described**	
LookupCCVNumber	Section 2.5.1.1.9.	
FQueue_Retrieve	Section 2.21.1.13.7.	
GetOldStyleCompany	Section 2.21.1.25.6.	
ATPPut	Section 2.21.1.7.4.	
CancelAlarm	Section 2.21.2.4.3.	
SetAlarm	Section 2.21.2.4.2.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.5.1.1.2.	

Table 2.5-20: ServiceRequest Information.

2.5.3.1.2 PerformRepair

PerformRepair is called when a repair has been completed. The function call is PerformRepair(t). Table 2.5-21 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to MaintRepairRequest	/simnet/mcc/include/Maint.h
state	register pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
capabilities	VehicleCapabilities	basic.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
TurnFleetIntoCapabilities	Section 2.5.2.1.7.	
LookupTruckNumber	Section 2.5.1.1.8.	
SendConsoleResponse	Section 2.21.1.7.6.	
SeeMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	

Called By	
Function	**Where Described**
RequestArrived	Section 2.5.1.1.4.

Table 2.5-21: PerformRepair Information.

2.5.3.1.3 ReadyTimeout

ReadyTimeout returns a truck to the ready state when no additional Service Request PDUs seem to be forthcoming. *param* contains the TruckState of the truck offering the supplies; *a* is the alarm which initiated this call. The function call is ReadyTimeout(param, a). Table 2.5-22 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
param	long	Standard C type.
a	Alarm	/simnet/rcs/libipc/alarm.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
state	pointer to TruckState	/simnet/mcc/Maint/Mainthost.h
req	MaintServicingRequest	/simnet/mcc/incluce/Maint.h
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	

Table 2.5-22: ReadyTimeout Information.

2.6 The FSE (Fire Support Element) Process

The FSE Process supports indirect fire missions carried out by mortars and howitzers. It controls the delivery of fire and controls when the mortars and howitzers are visible or invisible. Its structure is shown in Figure 2.6-1.

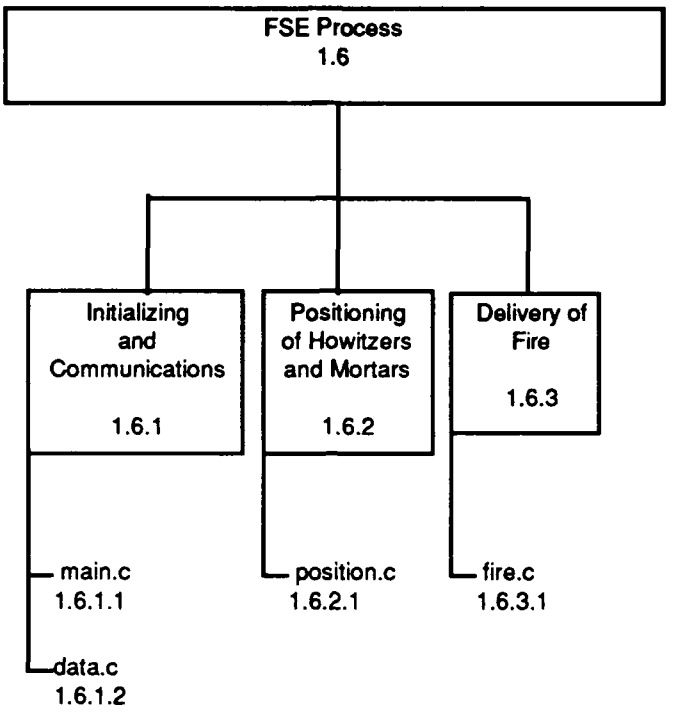


Figure 2.6-1: FSE (Fire Support Element) Process Structure.

The FSE process sends and receives the interprocess messages listed in Table 2.6-1.

Interprocess Message	Received From	Sent To	In Response To	Data Sent	Transaction Type
ShowCCV		Mother	CCV needs to be visible	-ownership -identifier -type -alignment -role -company -location -direction	First part of two-way interaction
HideCCV		Mother	CCV needs to be invisible	- indentifier	One-way interaction
KilledCCV	Mother		When CCV killed as shown by these PDUs: VehicleImpact Collision IndirectFire	- identifier	One-way interaction
ActivateResult	Mother		Placed combat vehicle or CCV.	- identifier - result	Second part of two-way interaction ShowCCV
FireVolley		Mother	Artillery shells detonate on or above battlefield	- type of ammo - type of fuze - flight time - number of shells For each shell's detonation: - horizontal location - height - delay before next	One-way interaction
ArtyDispatch	SCC		BattleMaster wants to dispatch platoon of howitzers	- identifier - platoon	One-way interaction
ArtyArrive	SCC		Howitzers have arrived	- identifier - platoon - location - azimuth	One-way interaction
Arty Reconstitute	SCC		BattleMaster wants to reconstitute platoon of howitzer battery	- identifier - platoon - location - supply quantities	One-way interaction

Table 2.6-1: FSE Process Interprocess Messages.

The FSE Process is composed of the following second level CSCs:

Initialization and Communication
Positioning of Howitzers and Mortars
Delivery of Fire

2.6.1 Initialization and Communication

The FSE Process opens a connection with the FSE Mac and sends down the initial location of the batteries of howitzers and platoon of mortars. It listens to messages from the Mother process such as those which describe when a gun is destroyed. It listens to messages from the FSE Console determining when a mortar is displaced, or when a fire mission is to be started. It consists of the following CSUs:

main.c
data.c

2.6.1.1 main.c
/simnet/mcc/FSE/main.c

main.c contains the top-level code for the FSE host process.

2.6.1.1.1 main

main is the entry point to the FSE process. The function call is main(argc, argv). Table 2.6-2 describes the parameters usedd and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.
Calls		
Function	**Where Described**	
InitializeProcess	Section 2.21.1.19.1.	
OpenHostSocket	Section 2.21.1.7.1.	
ActivateConsole	Section 2.21.1.7.2.	
AlarmsEnabled	Section 2.21.2.4.4.	

Table 2.6-2: main Information.

2.6.1.1.2 ProcessMessage

ProcessMessage processes a message from another process. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the length of the data in the message received. *data* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. The function call is ProcessMessage(type, kind, length, data). Table 2.6-3 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
ty pe	int	Standard C type.
king	long	Standard C type.
length	int	Standard C type.
data	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
alarmsEnabled	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESS AGE	Message of unexpected kind.	
Calls		
Function	**Where Described**	
AlarmsEnabled	Section 2.21.2.4.4.	
ATPTransact	Section 2.21.1.7.3.	
CloseSocket	Section 2.21.2.10.4.	
ResetClock	Section 2.6.1.1.7.	
TubeKilled	Section 2.6.1.1.6.	
ReconstituteBattery	Section 2.6.1.1.5.	
RemoteDispatch	Section 2.6.2.1.5.	
RemoteArrive	Section 2.6.2.1.6.	

Table 2.6-3: ProcessMessage Information.

2.6.1.1.3 RequestArrived

RequestArrived processes a transaction request, *t*, from the FSE Console. The function call is RequestArrived(t). Table 2.6-4 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	short	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_GETREQUEST_FAILE D	ATPGetRequest failed.	
MCC_UNKNOWN_REQUEST	Unkonwn request from Macintosh.	

Calls	
Function	**Where Described**
ATPGetRequest	Section 2.21.1.1.5.
ProcessGridInfoRequest	Section 2.21.1.16.1.
InitializeBattery	Section 2.6.1.1.4.
ATPTransact	Section 2.21.1.7.3.
FireRequest	Section 2.6.3.1.1.
FPFRequest	Section 2.6.3.1.4.
LocalDispatchRequest	Section 2.6.2.1.3.
LocalArriveRequest	Section 2.6.2.1.4.

Table 2.6-4: RequestArrived Information.

2.6.1.1.4 InitializeBattery

InitializeBattery initializes a battery of guns. The function call is InitializeBattery(battery, location, azimuth, ammo). Table 2.6-5 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
location	pointer to LongPt	/simnet/common/include/global/longpt.h
azimuth	int	Standard C type.
ammo	array of short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
b	register pointer to Battery	/simnet/mcc/FSE/FSEhost.h
t	register pointer to Tube	/simnet/mcc/FSE/FSEhost.h
i	short	Standard C type.
j	short	Standard C type.
n	short	Standard C type.
req	FSEGunInitRequest	/simnet/mcc/include/FSE.h
Calls		
Function	**Where Described**	
PlaceHalf	Section 2.6.2.1.2.	
ATPTransact	Section 2.21.1.7.3.	
Called By		
Function	**Where Described**	
ReconstituteBattery	Section 2.6.1.1.5.	
RequestArrived	Section 2.6.1.1.3.	

Table 2.6-5: InitializeBattery Information.

2.6.1.1.5 ReconstituteBattery

ReconstituteBattery is called to reset a battery to the ready state. *msg* contains detailed information about the battery. The function call is ReconstituteBattery(msg). Table 2.6-6 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	register pointer to MCCMessageBuffer	/simnet/mc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
battery	short	Standard C type.
i	short	Standard C type.
Calls		
Function	**Where Described**	
InitializeBattery	Section 2.6.1.1.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.6.1.1.2.	

Table 2.6-6: ReconstituteBattery Information.

2.6.1.1.6 TubeKilled

TubeKilled is called when a gun has been destroyed. The function call is TubeKilled(msg). Table 2.6-7 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	FSEKilledRequest	/simnet/mcc/include/FSE.h
i	short	Standard C type.
j	short	Standard C type.
k	short	Standard C type.
Calls		
Function	**Where Described**	
ATPPut	Section 2.21.1.7.4.	

Called By	
Function	**Where Described**
ProcessMessage	Section 2.6.1.1.2.

Table 2.6-7: TubeKilled Information.

2.6.1.1.7 ResetClock

ResetClock tells the Macintosh the time. The function call is ResetClock(). Table 2.6-8 describes the internal variables used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	FSEClockRequest	/simnet/mcc/include/FSE.h
Calls		
Function	**Where Described**	
MacintoshTime	Section 2.21.1.27.1.	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.6.1.1.2.	

Table 2.6-8: ResetClock Information.

2.6.1.2 data.c

/simnet/mcc/FSE/data.c

data.c contains data declarations for the FSE host process. Table 2.6-9 shows the varibles declared in data.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
batteries	array of Battery	/simnet/mcc/FSE/FSEhost.h

Table 2.6-9: data.c Variable Information.

2.6.2 Positioning of Howitzers and Mortars

The mortar platoon can be displaced from the FSE Macintosh while the howitzer batteries may only be moved by the Battlemaster via the SCC process. This file contains routines that control the howitzers and mortars visibility. The mortar platoon and howitzer batteries can be placed one half at a time. There is one CSU in this second level CSC, position.c.

2.6.2.1 position.c
/simnet/mcc/FSE/position.c

position.c contains routines which implement the positioning of batteries, including displacement.

2.6.2.1.1 IsMortar

IsMortar is a utility routine which determines whether *vehType* is a mortar. The function call is IsMortar(vehType). Table 2.6-10 describes the parameters used by using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehType	ObjectType	/simnet/common/include/protocol/basic.h
Return Values		
Return Value	**Type**	**Meaning**
TRUE	int	Vehicle is mortar.
FALSE	int	Vehicle is not mortar.
Called By		
Function	**Where Described**	
PlaceHalf	Section 2.6.2.1.2.	

Table 2.6-10: IsMortar Information.

2.6.2.1.2 PlaceHalf

PlaceHalf positions one half of a battery. The function call is PlaceHalf(b, half, location, azimuth). Table 2.6-11 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	pointer to Battery	/simnet/mcc/FSE/FSEhost.h
half	int	Standard C type.
location	pointer to LongPt	/simnet/common/include/global/longpt.h
azimuth	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
h	register pointer to Half	/simnet/mcc/FSE/FSEhost.h
t	pointer to Tube	/simnet/mcc/FSE/FSEhost.h
rad	float	Standard C type.
i	short	Standard C type.
pt	LongPt	/simnet/common/include/global/longpt.h
alignment	unsigned char	Standard C type.
marking	VehicleMarking	/simnet/common/include/protocol/basic.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
unit	OrganizationalUnit	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
mil_to_rad	Macro defined in /simnet/common/include/global/sim_macros.h.	
IsMortar	Section 2.6.2.1.1.	
GetGuises	Section 2.21.1.17.1.	
InitUnit	Section 2.21.1.19.4.	
DisplayCCV	Section 2.21.1.6.1.	
Called By		
Function	**Where Described**	
LocalArriveRequest	Section 2.6.2.1.4.	
RemoteArrive	Section 2.6.2.1.6.	
InitializeBattery	Section 2.6.1.2.4.	

Table 2.6-11: PlaceHalf Information.

2.6.2.1.3 LocalDispatchRequest

LocalDispatchRequest processes request, *t*, from the FSE Macintosh to dispatch guns. The function call is LocalDispatchRequest(t). Table 2.6-12 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
h	register pointer to Half	/simnet/mcc/FSE/FSEhost.h
i	short	Standard C type.
tubes	short	Standard C type.
dispatch	FSEDispatchRequest	/simnet/mcc/include/FSE.h

Calls	
Function	**Where Described**
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.
SendConsoleResponse	Section 2.21.1.7.6.
HideCCV	Section 2.21.1.6.2.
Called By	
Function	**Where Described**
RequestArrived	Section 2.6.1.1.3.

Table 2.6-12: LocalDispatchRequest Information.

2.6.2.1.4 LocalArriveRequest

LocalArriveRequest processes a request, *t*, from the FSE Macintosh to place a fire unit on the terrain. The function call is LocalArriveRequest(t). Table 2.6-13 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
arrive	FSEArriveRequest	/simnet/mcc/include/FSE.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
PlaceHalf	Section 2.6.2.1.2.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.6.1.1.3.	

Table 2.6-13: LocalArriveRequest Information.

2.6.2.1.5 RemoteDispatch

RemoteDispatch processes a notice from the SCC, *msg*, that some howitzers have begun to displace. The function call is RemoteDispatch(msg). Table 2.6-14 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	register pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
h	register pointer to Half	/simnet/mcc/FSE/FSEhost.h
tubes	short	Standard C type.
i	short	Standard C type.
req	FSEDispatchRequest	/simnet/mcc/include/FSE.h
Calls		
Function	**Where Described**	
HideCCV	Section 2.21.1.6.2.	
ATPTransact	Section 2.21.1.7.3.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.6.1.1.2.	

Table 2.6-14: RemoteDispatch Information.

2.6.2.1.6 RemoteArrive

RemoteArrive processes a notice from the SCC, *msg*, that some howitzers have arrived at their destination. The function call is RemoteArrive(msg). Table 2.6-15 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	register pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	FSEArriveRequest	/simnet/mcc/include/FSE.h
Calls		
Function	**Where Described**	
PlaceHalf	Section 2.6.2.1.2.	
ATPTransact	Section 2.21.1.7.3.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.6.1.1.2.	

Table 2.6-15: RemoteArrive Information.

2.6.3 Delivery of Fire

There are three different types of firing: Adjust Fire, Fire For Effect, and Final Protective Fire. All three types are handled by routines in fire.c. It provides routines that randomly distribute the bursts in the area designated and decrements the count of ammo from each of the simulated gun tubes. It sends Mother the schedule of fire so that Mother can then send out the appropriate packets on the network. There is one CSU, fire.c.

2.6.3.1 fire.c
/simnet/mcc/FSE/fire.c

fire.c contains routines which implement the Adjust Fire, Fire For Effect and FPF operations.

2.6.3.1.1 FireRequest

FireRequest processes a fire request, *t*, from the Macintosh. The function call is FireRequest(t). Table 2.6-16 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
fire	FSEFireRequest	/simnet/mcc/include/FSE.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
AdjustFire	Section 2.6.3.1.2.	
FireForEffect	Section 2.6.3.1.3.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.6.1.1.3.	

Table 2.6-16: FireRequest Information.

2.6.3.1.2 AdjustFire

AdjustFire fires an adjustment round from a battery. The function call is AdjustFire(fire). Table 2.6-17 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
fire	register pointer to FSEFireRequest	/simnet/mcc/include/FSE.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
b	pointer to Battery	/simnet/mcc/FSE/FSEhost.h
i	short	Standard C type.
j	short	Standard C type.
msg	MCCMesageBuffer	/simnet/mcc/include/MCC_ipc.h

Calls	
Function	**Where Described**
MapAmmoCode	Section 2.6.3.1.5.
TimeOfFlight	Section 2.6.3.1.6.
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.
UpdateSharedMemory	Section 2.6.3.1.11.
Called By	
Function	**Where Described**
FireRequest	Section 2.6.3.1.1.

Table 2.6-17: AdjustFire Information.

2.6.3.1.3 FireForEffect

FireForEffect fires a volley from a battery in a parallel sheaf. The function call is FireForEffect(fire, half). Table 2.6-18 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
fire	register pointer to FSEFireRequest	/simnet/mcc/include/FSE.h
half	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
b	pointer to Battery	/simnet/mcc/FSE/FSEhost.h
msg	MCCMesageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
MapAmmoCode	Section 2.6.3.1.5.	
TimeOfFlight	Section 2.6.3.1.6.	
PlaceParallelSheaf	Section 2.6.3.1.7.	
DistributeBursts	Section 2.6.3.1.8.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
UpdateSharedMemory	Section 2.6.3.1.11.	
Called By		
Function	**Where Described**	
FireRequest	Section 2.6.3.1.1.	

Table 2.6-18: FireForEffect Information.

2.6.3.1.4 FPFRequest

FPFRequest processes an FPF request, *t*, from the Macintosh. The function call is FPFRequest(t). Table 2.6-19 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
det	register pointer to struct detonation	
fpf	FSEFPFRequest	/simnet/mcc/include/FSE.h
b	pointer to Battery	/simnet/mcc/FSE/FSEhost.h
h	pointer to Half	/simnet/mcc/FSE/FSEhost.h
i	short	Standard C type.
j	short	Standard C type.
incX	float	Standard C type.
incY	float	Standard C type.
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc .h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
DistributeBursts	Section 2.6.3.1.8.	
MapAmmoCode	Section 2.6.3.1.5.	
TimeOfFlight	Section 2.6.3.1.6.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
UpdateSharedMemory	Section 2.6.3.1.11.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.6.1.1.3.	

Table 2.6-19: FPFRequest Information.

2.6.3.1.5 MapAmmoCode

MapAmmoCode maps the ammo codes used on the FSE Macintosh to those used on the network. The function call is MapAmmoCode(ammo, battery, msg). Table 2.6-20 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
ammo	short	Standard C type.
battery	pointer to Battery	/simnet/mcc/FSE/FSEhost.h
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc .h

Called By	
Function	**Where Described**
FPFRequest	Section 2.6.3.1.4.
FireForEffect	Section 2.6.3.1.3.
AdjustFire	Section 2.6.3.1.2.

Table 2.6-20: MapAmmoCode Information.

2.6.3.1.6 TimeOfFlight

TimeOfFlight returns the time of flight, in seconds, for a round to reach a target location from a particular battery. The function call is TimeOfFlight(b, half, location, fuze). Table 2.6-21 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	pointer to Battery	/simnet/mcc/FSE/FSEhost.h
half	int	Standard C type.
location	pointer to LongPt	/simnet/common/include/global/longpt.h
fuze	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pt	pointer to LongPt	/simnet/common/include/global/longpt.h
t1	float	Standard C type.
t2	float	Standard C type.
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
b->timeOfFlight	int	Time of flight for round.
Called By		
Function	**Where Described**	
FPFRequest	Section 2.6.3.1.4.	
FireForEffect	Section 2.6.3.1.3.	
AdjustFire	Section 2.6.3.1.2.	

Table 2.6-21: TimeOfFlight Information.

2.6.3.1.7 PlaceParallelSheaf

PlaceParallelSheaf places bursts in a parallel sheaf. The function call is PlaceParallelSheaf(b, half, source, dest, msg). Table 2.6-22 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	pointer to Battery	/simnet/mcc/FSE/FSEhost.h
half	int	Standard C type.
source	pointer to LongPt	/simnet/common/include/global/longpt.h
dest	pointer to LongPt	/simnet/common/include/global/longpt.h
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
t	register pointer to Tube	/simnet/mcc/FSE/FSEhost.h
det	register pointer to struct detonation	
displace	array 2 of float	Standard C type.
i	short	Standard C type.
j	short	Standard C type.
Called By		
Function	**Where Described**	
FireForEffect	Section 2.6.3.1.3.	

Table 2.6-22: PlaceParallelSheaf Information.

2.6.3.1.8 DistributeBursts

DistributeBursts randomly distributes bursts in time and space. The function call is DistributeBursts(msg, fuze, first). Table 2.6-23 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	register pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
fuze	int	Standard C type.
first	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
det	register pointer to struct detonation	
Calls		
Function	**Where Described**	
RandomDistance	Section 2.6.3.1.9.	
RandomDelay	Section 2.6.3.1.10.	

Called By	
Function	**Where Described**
FPFRequest	Section 2.6.3.1.4.
FireForEffect	Section 2.6.3.1.3.

Table 2.6-23: DistributeBursts Information.

2.6.3.1.9 RandomDistance

RandomDistance generates a random distance abberation. The function call is RandomDistance(). Table 2.6-24 describes the internal variables used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
i	float	Random distance abberation.
Called By		
Function	**Where Described**	
DistributeBursts	Section 2.6.3.1.8.	

Table 2.6-24: RandomDistance Information.

2.6.3.1.10 RandomDelay

RandomDelay generates a random time abberation. The function call is RandomDelay(). Table 2.6-25 describes the value returned by this function.

Return Values		
Return Value	**Type**	**Meaning**
rand() & 0x01C0	int	Random time delay.
Called By		
Function	**Where Described**	
DistributeBursts	Section 2.6.3.1.8.	

Table 2.6-25: RandomDelay Information.

2.6.3.1.11 UpdateSharedMemory

UpdateSharedMemory updates the record of ammo supply maintained in shared memory. The function call is UpdateSharedMemory(battery, fuze). Table 2.6-26 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
fuze	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
b	register pointer to Batttery	/simnet/mcc/FSE/FSEhost.h
i	short	Standard C type.

Called By	
Function	**Where Described**
FPFRequest	Section 2.6.3.1.4.
FireForEffect	Section 2.6.3.1.3.
AdjustFire	Section 2.6.3.1.2.

Table 2.6-26: UpdateSharedMemory Information.

2.7 The CAS (Close Air Support) Process

The CAS process is responsible for conveying information about close air support missions from the CAS Macintosh to the outside world. The structure is shown in Figure 2.7-1.

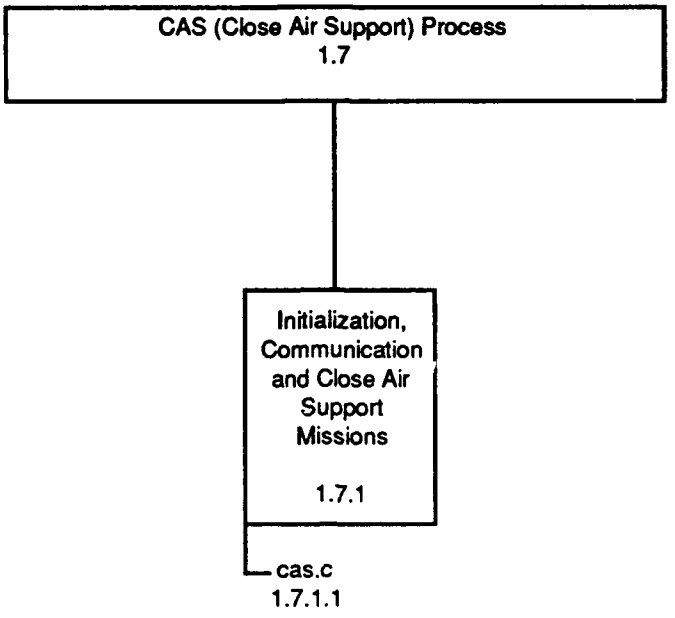


Figure 2.7-1: SCC (SIMNET Control Console) Process Structure.

Table 2.7-1 shows the interprocess messages that are used by the CAS process.

Interprocess Message	Received From	Sent To	In Response To	Data Sent	Transaction Type
PlaceBombs		Mother	When sortie drops bombs on battlefield	- target location - run-in heading	One-way interaction
Sorties	SCC		BattleMaster has allocated sorties	- number of sorties - number of sorties that may be preplanned	One-way interaction

Table 2.7-1: CAS Interprocess Message Information.

The CAS process is composed of the Initialization and Communication second level CSC.

2.7.1 Initialization and Communication

The CAS Process is initialized and told the number of total sorties allowed and the number of sorties allowed to be preplanned. It acts as a liaison between the CAS Macintosh and the Mother process. If more sorties are added by the Battlemaster, the CAS process conveys this new information to the CAS Console.

The scheduling of missions is implemented entirely within the CAS Macintosh Console. Only when a mission is "cleared hot" is the CAS process notified of the mission. The CAS process then uses the information given to determine the location of the individual detonations on the battlefield. It looks to see if there are any combat vehicles close to the given location and adjusts the run-in heading and location of the bursts accordingly, to try to be more representative of how a real mission would be adjusted. When a sortie is to drop a bomb, the CAS process notifies the Mother process to report the detonation on the SIMNET local area network. There is one CSU, cas.c.

2.7.1.1 cas.c
/simnet/mcc/CAS/cas.c

cas.c contains the CAS process, which manages the CAS console.

2.7.1.1.1 main

main is the entry point to the CAS process. The function call is main(argc, argv). Table 2.7-2 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
basic.hbasic.hargv	pointer to pointer to char	Standard C type.
Calls		
Function	**Where Described**	
InitializeProcess	Section 2.21.1.19.1.	
OpenHostSocket	Section 2.21.1.7.1.	
ActivateConsole	Section 2.21.1.7.2.	
AllotSorties	Section 2.7.1.1.4.	
AlarmsEnabled	Section 2.21.2.4.4.	

Table 2.7-2: main Information.

2.7.1.1.2 ProcessMessage

ProcessMessage processes a message from another process. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the length of the data in the message received. *data* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. The function call is ProcessMessage(type, king, length, data). Table 2.7-3 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	long	Standard C type.
length	int	Standard C type.
basic.hdata	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
alarmsEnabled	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESSAGE	Message of unexpected kind.	
Calls		
Function	**Where Described**	
AlarmsEnabled	Section 2.21.2.4.4.	
ATPTransact	Section 2.21.1.7.3.	
CloseSocket	Section 2.21.2.10.4.	
AllotSorties	Section 2.7.1.1.4.	
ResetClock	Section 2.7.1.1.7.	

Table 2.7-3: ProcessMessage Information.

2.7.1.1.3 RequestArrived

RequestArrived processes a transaction request, *t*, from the CAS console. The function call is RequestArrived(t). Table 2.7-4 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
basic.ht	pointer to Transaction	/simnet/mcc/include/bridge.h
Errors		
Error Name	**Reason for Error**	
MCC_GETREQUEST_FAILED	ATPGetRequest failed.	
MCC_UNKNOWN_REQUEST	Unknown request from Macintosh.	
Calls		
Function	**Where Described**	
ATPGetRequest	Section 2.21.1.1.5.	
ProcessGridInfoRequest	Section 2.21.1.16.1.	
AllotSorties	Section 2.7.1.1.4.	
BombRequest	Section 2.7.1.1.5.	

Table 2.7-4: RequestArrived Information.

2.7.1.1.4 AllotSorties

AllotSorties tells the Macintosh of sorties allocated for today. The function call is AllotSorties(). Table 2.7-5 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	CASSortiesRequest	/simnet/mcc/include/CAS.h
Calls		
Function	**Where Described**	
ATPPut	See console.c in Section 2.21.1.7.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.7.1.1.3.	
ProcessMessage	Section 2.7.1.1.2.	
main	Section 2.7.1.1.1.	

Table 2.7-5: AllotSorties Information.

2.7.1.1.5 BombRequest

BombRequest processes a bombing request, *t*, from the CAS console. The function call is BombRequest(t). Table 2.7-6 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
basic.ht	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
basic.hreq	pointer to CASBombRequest	/simnet/mcc/include/CAS.h
basic.hmission	pointer to Mission	/simnet/mcc/CAS/cas.c
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
deg_to_rad	Macro defined in simnet/common/include/global/sim_macros.h.	
DropBombs	Section 2.7.1.1.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.7.1.1.3.	

Table 2.7-6: BombRequest Information.

2.7.1.1.6 DropBombs

DropBombs is called when it is time for a sortie to drop its bombs. The function call is DropBombs(param, a). Table 2.7-7 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
param	long	Standard C type.
a	Alarm	/simnet/rcs/libipc/alarm.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
basic.hmission	pointer to Mission	/simnet/mcc/CAS/cas.c
Calls		
Function	**Where Described**	
SendMCCMsg1	Macro defined in /simnet/mcc/include/bridge.h.	
SetAlarm	Section 2.21.2.4.2.	
Called By		
Function	**Where Described**	
BombRequest	Section 2.7.1.1.5.	

Table 2.7-7: DropBombs Information.

2.7.1.1.7 ResetClock

ResetClock tells the CAS Console Macintosh the time. The function call is ResetClock(). Table 2.7-8 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	CASClockRequest	/simnet/mcc/include/CAS.h
Calls		
Function	**Where Described**	
MacintoshTime	Section 2.21.1.27.1.	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.7.1.1.2.	

Table 2.7-8: ResetClock Information.

2.8 The CEW (Combat Engineering Workstation) Process

The Combat Engineering Workstation process is responsible for the emplacement and breaching of minefields. The structure is shown in Figure 2.8-1.

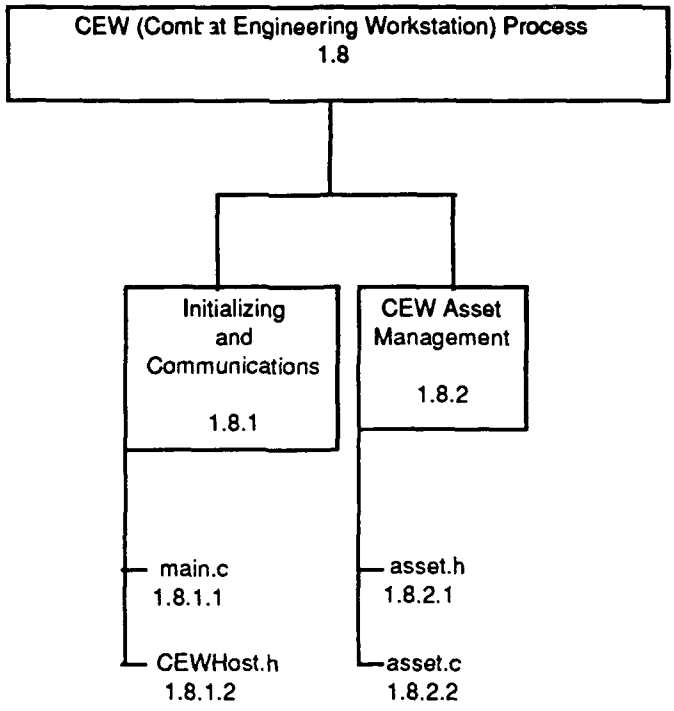


Figure 2.8-1: CEW (Combat Engineering Workstation) Process Structure.

2.8.1 Initialization and Communication

The CEW Process opens a connection with the CEW Macintosh and then sends down the initial location of all minefields. It listens to messages from the Mother process and passes information along to the Macintosh. It contains one CSU, main.c.

2.8.1.1 main.c

/simnet/mcc/CEW/main.c

main.c contains routines for the Immediate Mine Emplacement (CEW) host process.

2.8.1.1.1 main

main is the entry point to the CEW process. The function call is main(argc, argv). Table 2.8-1 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.
Calls		
Function	**Where Described**	
InitializeProcess	Section 2.21.1.19.1.	
OpenHostSocket	Section 2.21.1.7.1.	
ActivateConsole	Section 2.21.1.7.2.	
AlarmsEnabled	Section 2.21.2.4.4.	

Table 2.8-1: main Information.

2.8.1.1.2 ProcessMessage

ProcessMessage process a message from another host process. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the length of the data in the message received. *data* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. The function call is ProcessMessage(type, kind, length, data). Table 2.8-2 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	long	Standard C type.
length	int	Standard C type.
data	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
alarmsEnabled	int	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_UNEXPECTED_MESSAGE	Message of an unexpected type.	
Calls		
Function	**Where Described**	
AlarmsEnabled	Section 2.21.2.4.4.	
ATPTransact	Section 2.21.1.7.3.	
CloseSocket	Section 2.21.2.10.4.	
ResetClock	Section 2.8.1.1.3.	
AssetReconstituted	Section 2.8.2.2.4.	
AssetKilled	Section 2.8.2.2.5.	

Table 2.8-2: ProcessMessage Information.

2.8.1.1.3 ResetClock

ResetClock updates the clock on the CEW Console Macintosh. The function call is ResetClock(). Table 2.8-3 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	CEWClockRequest	/simnet/mcc/include/CEW.h
Calls		
Function	**Where Described**	
MacintoshTime	Section 2.21.1.27.1.	
ATPPut	Section 2.21.1.7.4.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.8.1.1.2.	

Table 2.8-3: ResetClock Information.

2.8.1.1.4 RequestArrived

RequestArrived processes an ATP Request, *t*, from the CEW Console Macintosh. The function call is RequestArrived(t). Table 2.8-4 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Errors		
Error Name	**Reason for Error**	
MCC_GETREQUEST_FAILED	ATPGetRequest failed.	
MCC_UNKNOWN_REQUEST	Unknown request from Macintosh.	
Calls		
Function	**Where Described**	
ATPGetRequest	Section 2.21.1.1.5.	
ProcessGridInfoRequest	Section 2.21.1.16.1.	
InitializeAssets	Section 2.8.2.2.2.	
ProcessBoundsRequest	Section 2.8.1.1.5.	
ProcessEmplaceRequest	Section 2.8.1.1.6.	
ProcessBreachRequest	Section 2.8.1.1.7.	
ProcessSupplyRequest	Section 2.8.1.1.10.	
ProcessDispatchRequest	Section 2.8.1.1.8.	
ProcessArrivalRequest	Section 2.8.1.1.9.	

Table 2.8-4: RequestArrived Information.

2.8.1.1.5 ProcessBoundsRequest

ProcessBoundsRequest handles a request from the CEW Console Macintosh for information about terrain boundaries. The function call is ProcessBoundsRequest(t). Table 2.8-5 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
rsp	register pointer to CEWBoundsResponse	/simnet/mcc/include/CEW.h
Calls		
Function	**Where Described**	
ATPResponseData	Macro defined in /simnet/mcc/include/bridge.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.8.1.1.4.	

Table 2.8-5: ProcessBoundsRequest Information.

2.8.1.1.6 ProcessEmplaceRequest

ProcessEmplaceRequest handles a request from the CEW Console Macintosh to emplace a minefield. The function call is ProcessEmplaceRequest(t). Table 2.8-6 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to CEWEmplaceRequest	/simnet/mcc/includ/CEW.h
rsp	pointer to CEWEmplaceResponse	/simnet/mcc/includ/CEW.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
ATPResponseData	Macro defined in /simnet/mcc/include/bridge.h.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
SendConsoleResponse	Section 2.21.1.7.6.	

Called By	
Function	**Where Described**
RequestArrived	Section 2.8.1.1.4.

Table 2.8-6: ProcessEmplaceRequest Information.

2.8.1.1.7 ProcessBreachRequest

ProcessBreachRequest handles a request, *t*, from the CEW Console Macintosh to breach a minefield. The function call is ProcessBreachRequest(t). Table 2.8-7 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to CEWBreachRequest	/simnet/mcc/includ/CEW.h
rsp	pointer to CEWBreachResponse	/simnet/mcc/includ/CEW.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
ATPResponseData	Macro defined in /simnet/mcc/include/bridge.h.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.8.1.1.4.	

Table 2.8-7: ProcessBreachRequest Information.

2.8.1.1.8 ProcessDispatchRequest

ProcessDispatchRequest handles a request, *t*, from the CEW Console Macintosh to hide a CCV vehicle being dispatched. The function call is ProcessDispatchRequest(t). Table 2.8-8 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to CEWDispatchRequest	/simnet/mcc/includ/CEW.h
rsp	pointer to CEWDispatchResponse	/simnet/mcc/includ/CEW.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
ATPResponseData	Macro defined in /simnet/mcc/include/bridge.h.	
HideCCV	Section 2.21.1.6.2.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.8.1.1.4.	

Table 2.8-8: ProcessDispatchRequest Information.

2.8.1.1.9 ProcessArrivalRequest

ProcessArrivalRequest handles a request from the CEW Console Macintosh to show an arrived CCV vehicle. The function call is ProcessArrivalRequest(t). Table 2.8-9 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to CEWArrivalRequest	/simnet/mcc/includ/CEW.h
rsp	pointer to CEWArrivalResponse	/simnet/mcc/includ/CEW.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
ccv	pointer to CCVStatus	/simnet/mcc/include/veh_table.h
marking	VehicleMarking	basic.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
ATPResponseData	Macro defined in /simnet/mcc/include/bridge.h.	
DisplayCCV	Section 2.21.1.6.1.	
SendConsoleResponse	Section 2.21.1.7.6.	

Called By	
Function	**Where Described**
RequestArrived	Section 2.8.1.1.4.

Table 2.8-9: ProcessArrivalRequest Information.

2.8.1.1.10 ProcessSupplyRequest

ProcessSupplyRequest handles a request from the CEW Console Macintosh to supply a CEW. The function call is ProcessSupplyRequest(t). Table 2.8-10 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
t	pointer to Transaction	/simnet/mcc/include/bridge.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	pointer to CEWSupplyRequest	/simnet/mcc/includ/CEW.h
rsp	pointer to CEWSupplyResponse	/simnet/mcc/includ/CEW.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
temp	CEWSupplyRequest	/simnet/mcc/includ/CEW.h
Calls		
Function	**Where Described**	
ATPRequestData	Macro defined in /simnet/mcc/include/bridge.h.	
ATPResponseData	Macro defined in /simnet/mcc/include/bridge.h.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/MCC_ipc.h.	
SendConsoleResponse	Section 2.21.1.7.6.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.8.1.1.4.	

Table 2.8-10: ProcessSupplyRequest Information.

2.8.1.2 CEWHost.h

/simnet/mcc/CEW/CEWHost.h

CEWHost.h defines structures used by the CEW process. Table 2.8-11 describes the variables used by CEWHost.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
cewCCVState	extern array of TruckState	/simnet.mcc/CEW/CEWHost.h

Table 2.8-11: CEWHost.h Variable Information.

2.8.2 CEW Asset Management

2.8.2.1 asset.h
/simnet/mcc/CEW/asset.h

asset.h is the header file associated with asset.c. Section 2.8.2.2.

2.8.2.2 asset.c
/simnet/mcc/CEW/asset.c

asset.c contains routines for coordinating CEW assets. Table 2.8-12 describes the variables used by asset.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
assetTable	array of AssetEntry	/simnet/mcc/CEW/asset.c

Table 2.8-12: asset.c Variable Information.

2.8.2.2.1 LookupAssetNumber

LookupAssetNumber finds an asset record in the assetTable, given *kind* and *number*. The function call is LookupAssetNumber(kind, number). Table 2.8-13 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
kind	int	Standard C type.
number	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
assetTable[i].start + number	long	The asset record.
0	long	Error occured.
Called By		
Function	**Where Described**	
AssetReconstituted	Section 2.8.2.2.4.	

Table 2.8-13: LookupAssetNumber Information.

2.8.2.2.2 InitializeAssets

InitializeAssets is called when the CEW is initially spawned to read information in shared memory which has been initialized by the SCC. The SCC has already allocated the assets available to the CEW. InitializeAssets notifies the CEW Console Macintosh of all the resources which have been allocated. The function call is InitializeAssets(). Table 2.8-14 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
kind	int	Standard C type.
entry	int	Standard C type.
GetCEWParameters()	Function that returns pointer to CEWParameters	Section 2.21.1.8.1.
Calls		
Function	**Where Described**	
GetCEWParameters	Section 2.21.1.8.1.	
InitializeAsset	Section 2.8.2.2.3.	
ATPTransact	Section 2.21.1.7.3.	
Called By		
Function	**Where Described**	
RequestArrived	Section 2.8.1.1.4.	

Table 2.8-14: InitializeAssets Information.

2.8.2.2.3 InitializeAsset

InitializeAsset initializes one CEW asset by notifying the CEW Console Macintosh of its existence. *ccvNumber* is a unique ID for the vehicle (it is an index into the CCV truck array). *kind* is the kind of CEW CCV vehicle, and *pars* describes the CEW parameters for that asset. The function call is InitializeAsset(ccvNumber, kind, pars). Table 2.8-15 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
ccvNumber	int	Standard C type.
kind	short	Standard C type.
pars	pointer to CEWParameters	/simnet/mcc/include/MCC_pars.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	CEWAssetInitRequest	/simnet/mcc/include/CEW.h
guises	VehicleGuises	/simnet/common/include/protocol/basic.h
marking	VehicleMarking	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
ccv	pointer to CCVStatus	/simnet/mcc/include/veh_table.h

Calls	
Function	**Where Described**
DisplayCCV	Section 2.21.1.6.1.
ATPTransact	Section 2.21.1.7.3.
Called By	
Function	**Where Described**
InitializeAssets	Section 2.8.2.2.2.

Table 2.8-15: InitializeAsset Information.

2.8.2.2.4 AssetReconstituted

AssetReconstituted is called when the SCC notifies the CEW process that one of its CCVs has been reconstituted. The CEW process must notify the CEW Console Macintosh that this has occured via an assetInit message. The function call is AssetReconstituted(msg). Table 2.8-16 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	CEWAssetReconstRequest	/simnet/mcc/include/CEW.h
ccv	pointer to CCVStatus	/simnet/mcc/include/veh_table.h
pars	pointer to CEWParameters	/simnet/mcc/include/MCC_pars.h
marking	VehicleMarking	/simnet/common/include/protocol/basic.h
capabilities	VehicleCapabilities	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
LookupAssetNumber	Section 2.8.2.2.1.	
GetCEWParameters	Section 2.21.1.8.1.	
DisplayCCV	Section 2.21.1.6.1.	
ATPTransact	Section 2.21.1.7.3.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.8.1.1.2.	

Table 2.8-16: AssetReconstituted Information.

2.8.2.2.5 AssetKilled

AssetKilled is called when the Mother process notifies the CEW process that one of its CCVs has been killed. The CEW process must notify the CEW Console Macintosh that this has occured via an assetStatus message. The function call is AssetKilled(msg). Table 2.8-17 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	CEWAssetStatusRequest	/simnet/mcc/include/CEW.h
Calls		
Function	**Where Described**	
ATPTransact	Section 2.21.1.7.3.	
Called By		
Function	**Where Described**	
ProcessMessage	Section 2.8.1.1.2.	

Table 2.8-17: AssetKilled Information.

2.9 The Terminal Process

The Terminal Process acts as a simple front end to the MCC host system. It is intended only as a diagnostic tool and need not be used during normal operation of the MCC. The Terminal front end provides rudimentary querying of internal MCC states and data structures. The structure of this CSC is shown in Figure 2.9-1.

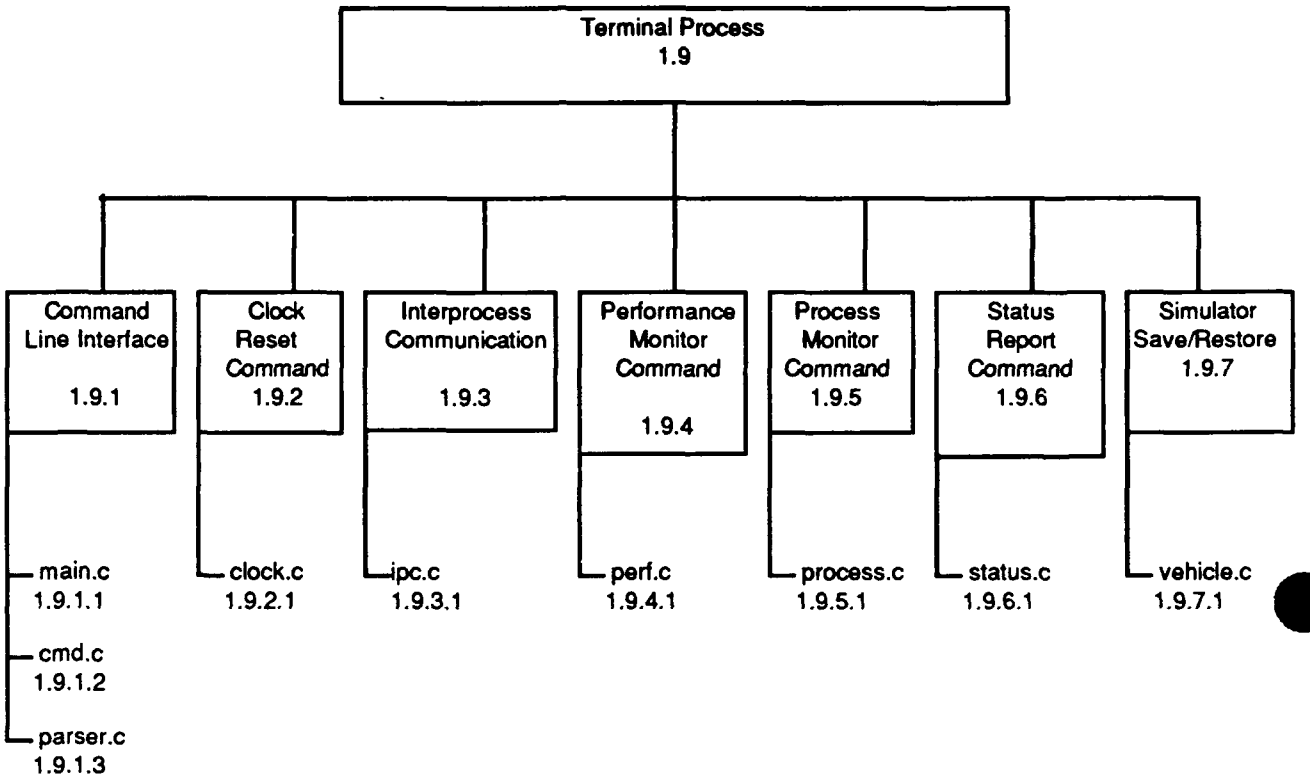


Figure 2.9-1: Terminal Process Structure.

The Terminal process consists of seven second level CSCs:

Command Line Interface
Clock Reset Command
Interprocess Communication
Performance Monitor Command
Process Monitor Command
Status Report Command
Simulator Save/Restore

2.9.1 Command Line Interface

The command line interface is a simple line-oriented interface. Commands are simple English phrases terminated by a new line. The CLI parses command lines and dispatches the appropriate routine to display the requested information. The CLI also contains code to send a restart request message to the Admin/Log, Maintenance, Fire Support, and Close Air Support, so that those Macintosh consoles may be restarted in the event of a power glitch or other failure. It contains three CSU's: main.c, cmd.c, and parser.c.

2.9.1.1 main.c
/simnet/mcc/Terminal/main.c

main.c contains most of the Terminal process. It accepts and processes commands from the MCC host's console terminal. Table 2.9-1 describes the variables used by main.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
Stopped	int	Standard C type.
terminal_disabled	int	Standard C type.
cardMemory	pointer to Card_Memory_t	/simnet/mcc/Terminal/main.c
done	int	Standard C type.
channel	int	Standard C type.

Table 2.9-1: main.c Variable Information.

2.9.1.1.1 Terminal_Handler

Terminal_Handler handles ^Z's typed at the keyboard. The function call is Terminal_Handler(). This function calls only standard functions and is not called by any other function in this CSC, so no table is included for it.

2.9.1.1.2 Quit_Handler

Quit_Handler is called when the MCC process is terminated. The function call is Quit_Handler(). Table 2.9-2 describes the functions called using this function.

Calls	
Function	**Where Described**
SendMCCMsg0	Macro defined in /simnet/mcc/include/bridge.h.

Table 2.9-2: Quit_Handler Information.

2.9.1.1.3 main

main is the entry point to the Terminal process. The function call is main(argc, argv). Table 2.9-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cp	register pointer to char	Standard C type.
saveCmdBuf	array 256 of char	Standard C type.
cmdBuf	array 256 of char	Standard C type.
cmd	array 256 of char	Standard C type.
params	array 256 of char	Standard C type.
par1	array 40 of char	Standard C type.
par2	array 40 of char	Standard C type.
status	char	Standard C type.
cardMemorySize	int	Standard C type.

Calls	
Function	**Where Described**
InitializeProcess	Section 2.21.1.19.1.
net_hostbuf_info	Section 2.20.2.6.1.
parser_init	Section 2.9.1.3.1.
tty_tick	libtty
parser_restore_term	Section 2.9.1.3.2.
CloseSocket	Section 2.21.2.10.4.

Table 2.9-3: main Information.

2.9.1.2 cmd.c

simnet/mcc/Terminal/cmd.c

cmd.c contains the command line interface dispatcher routines. Table 2.9-4 describes the variables used by cmd.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
terminal_disabled	int	Standard C type.
cardMemory	extern pointer to Card_Memory_t	simnet/mcc/Terminal/main.c
done	extern int	Standard C type.
networkInterface	extern int	Standard C type.

Table 2.9-4: cmd.c Variable Information.

2.9.1.2.1 cmd_send_breach

cmd_send_breach performs the command line action of creating a breached lane. The function call is cmd_send_breach(argc, argv). Table 2.9-5 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
msg	MCCMessageBuffer	/simnet/mcc/include/mcc_ipc.h
Calls		
Function	**Where Described**	
SendMCCMsg1	macro defined in /simnet/mcc/include/bridge.h	
Called By		
Function	**Where Described**	
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-5: cmd_send_breach Information.

2.9.1.2.2 cmd_clock

cmd_clock performs the command line action of changing the Macintosh system clocks. The function call is cmd_clock(argc, argv). Table 2.9-6 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
ClockCommand	Section 2.9.2.1.1.	
Called By		
Function	**Where Described**	
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-6: cmd_clock Information.

2.9.1.2.3 cmd_disable

cmd_disable performs the command line action of disabling the Terminal Input. The function call is cmd_disable(argc, argv). Table 2.9-7 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
parser_restore_term	Section 2.9.1.3.2.	

Called By	
Function	**Where Described**
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c

Table 2.9-7: cmd_disable Information.

2.9.1.2.4 cmd_exit

cmd_exit performs the command line action of exiting the MCC and deactivating any activated simulators. The function call is cmd_exit(argc, argv). Table 2.9-8 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv]	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
SendMCCMsg0	macro defined in /simnet/mcc/include/bridge.h	
Called By		
Function	**Where Described**	
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-8: cmd_exit Information.

2.9.1.2.5 cmd_list

cmd_list performs the command line action of listing the simulator save files in a given directory. The function call is cmd_clock(argc, argv). Table 2.9-9 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
SimListSaves	Section 2.9.7.1.1.	
Called By		
Function	**Where Described**	
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-9: cmd_list Information.

2.9.1.2.6 cmd_perf

cmd_perf performs the command line action of showing MCC performance data. The function call is cmd_perf(argc, argv). Table 2.9-10 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
ReportPerformance	Section 2.9.4.1.1.	
Called By		
Function	**Where Described**	
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-10: cmd_perf Information.

2.9.1.2.7 cmd_restart

cmd_restart performs the command line action of restarting a Macintosh console. The function call is cmd_restart(argc, argv). Table 2.9-11 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
console	register pointer to struct ConsoleName	
msg	MCCMessageBuffer	
Calls		
Function	**Where Described**	
SendMCCMsg1	macro defined in /simnet/mcc/include/bridge.h	
Called By		
Function	**Where Described**	
"MC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-11: cmd_restart Information.

2.9.1.2.8 cmd_restore

cmd_restore performs the command line action of restoring a simulator save file. The function call is cmd_restore(argc, argv). Table 2.9-12 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
SimRestoreAll	Section 2.9.7.1.3.	
Called By		
Function	**Where Described**	
"MCC Commands" top_command_file	/simnet/mcc/Terminal/parser.c	

Table 2.9-12: cmd_restore Information.

2.9.1.2.9 cmd_save

cmd_save performs the command line action of saving a simulator save file. The function call is cmd_save(argc, argv). Table 2.9-13 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
SimSaveAll	Section 2.9.7.1.2.	
Called By		
Function	**Where Described**	
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-13: cmd_save Information.

2.9.1.2.10 cmd_status_all

cmd_status_all provides command line status information about the MCC and network. The function call is cmd_status_all(argc, argv). Table 2.9-14 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
ReportIPCStatus	Section 2.9.3.1.1.	
net_print_statistics	Section 2.20.2.5.4.	
ProcessStatus	Section 2.9.5.1.1.	
SimStatusReport	Section 2.9.6.1.1.	
ReportYUMMStatus	Section 2.21.2.10.12	
Called By		
Function	**Where Described**	
"Status Commands" status_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-14: cmd_status_all Information.

2.9.1.2.11 cmd_status_bumper

cmd_status_bumper provides command line status information, specified by its bumper number, about the simulator. The function call is cmd_status_bumper(argc, argv). Table 2.9-15 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
VehFromBumper	Section 2.9.7.1.5.	
SimStatusReport	Section 2.9.6.1.1.	
Called By		
Function	**Where Described**	
"Status Commands" status_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-15: cmd_status_bumper Information.

2.9.1.2.12 cmd_status_ipc

cmd_status_ipc provides command line status information about the ipc parameters. The function call is cmd_restore(argc, argv). Table 2.9-16 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
ReportIPCStatus	Section 2.9.3.1.1.	
Called By		
Function	**Where Described**	
"Status Commands" status_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-16: cmd_status_ipc Information.

2.9.1.2.13 cmd_status_net

cmd_status_net provides command line status information about the simulation network. The function call is cmd_status_net(argc, argv). Table 2.9-17 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
net_print_statistics	Section 2.20.2.5.4	
Called By		
Function	**Where Described**	
"Status Commands" status_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-17: cmd_status_net Information.

2.9.1.2.14 cmd_status_process

cmd_status_process provides command line status information about the MCC processes. The function call is cmd_status_process(argc, argv). Table 2.9-18 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
ProcessStatus	Section 2.9.5.1.1	

Called By	
Function	**Where Described**
"Status Commands" status_command_table	/simnet/mcc/Terminal/parser.c

Table 2.9-18: cmd_status_process Information.

2.9.1.2.15 cmd_status_sim

cmd_status_sim provides command line status information about a specified simulator. The function call is cmd_restore(argc, argv). Table 2.9-19 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	SimulatorStatus	/simnet/mcc/include/veh_table.h
Calls		
Function	**Where Described**	
VehFromSimNum	Section 2.9.7.1.4.	
SimStatusReport	Section 2.9.6.1.1.	
Called By		
Function	**Where Described**	
"Status Commands" status_command_line	/simnet/mcc/Terminal/parser.c	

Table 2.9-19: cmd_status_sim Information.

2.9.1.2.16 cmd_status_sims

cmd_status_sims provides command line status information about this MCC's simulators. The function call is cmd_status_sims(argc, argv). Table 2.9-19 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.

Calls	
Function	**Where Described**
SimStatusReport	Section 2.9.6.1.1.
Called By	
Function	**Where Described**
"Status Commands" status_command_line	/simnet/mcc/Terminal/parser.c

Table 2.9-19: cmd_status_sims Information.

2.9.1.2.17 cmd_status_vid

cmd_status_vid provides command line status information about the simulator, specified by its vehicle ID number. The function call is cmd_status_vid(argc, argv). Table 2.9-20 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vehicleID	VehicleID	/simnet/common/include/protocol/basic.h
Calls		
Function	**Where Described**	
VehicleStatusReport	Section 2.9.6.1.2	
Called By		
Function	**Where Described**	
"Status Commands" status_command_line	/simnet/mcc/Terminal/parser.c	

Table 2.9-20: cmd_status_vid Information.

2.9.1.2.18 cmd_status_yumm

cmd_status_yumm provides command line status information about the YUMM internals. The function call is cmd_status_yumm(argc, argv). Table 2.9-21 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
ReportYUMMStatus	Section 2.21.2.10.12.	

Called By	
Function	**Where Described**
"Status Commands" status_command_line	/simnet/mcc/Terminal/parser.c

Table 2.9-21: cmd_status_yumm Information.

2.9.1.2.19 cmd_trace

cmd_trace performs the command line action of tracing network messages and internal messages. The function call is cmd_trace(argc, argv). Table 2.9-22 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to array of char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cp	register pointer to char	Standard C type.
Called By		
Function	**Where Described**	
"MCC Commands" top_command_table	/simnet/mcc/Terminal/parser.c	

Table 2.9-22: cmd_trace Information.

2.9.1.3 parser.c

simnet/mcc/Terminal/parser.c

parser.c is a command line interpreter definition file. It defines the commands available on the MCC host console. parser.c utilizes the libparser interface file to define the commands, options, and values which can be entered at the command line. Through libparser, the commands defined in parser.c are checked for validity, and the corresponding action routine in cmd.c is invoked. Table 2.9-23 describes the variables used by parser.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
top_command_table	extern array of PARSE_TABLE	parser.h (for libparser)

Table 2.9-23: parser.c Variable Information.

2.9.1.3.1 parser_init

parser_init intializes the parser. The function call is parser_init(). Table 2.9-24 describes the parameters used, errors returned and functions called using this function.

Calls	
Function	**Where Described**
tty_parser_init	libtty
Called By	
Function	**Where Described**
main	Section 2.9.1.1.3

Table 2.9-24: parser_init Information.

2.9.1.3.2 parser_restore_term

parser_restore_term restores the terminal process. The function call is parser_restore_term(). Table 2.9-25 describes the parameters used, errors returned and functions called using this function.

Calls	
Function	**Where Described**
tty_exit	libtty
Called By	
Function	**Where Described**
main	Section 2.9.1.1.3.
cmd_disable	Section 2.9.1.2.3.

Table 2.9-25: parser_restore_term Information.

2.9.2 Clock Reset Command

The Clock Reset Command allows an operator to change the Time-Of-Day clock on the MCC host system and all of the Macintosh Consoles. This is sometimes necessary due to normal clock skew so that operations among the MCC components will remain synchronized. This functionality is realized by one CSU, clock.c.

2.9.2.1 clock.c

simnet/mcc/Terminal/clock.c

clock.c contains functions which control the MCC clocks. Table 2.9-26 describes the variables used by clock.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
month_to_day	array 12 of int	Standard C type.

Table 2.9-26: clock.c Variable Information.

2.9.2.1.1 ClockCommand

ClockCommand resets the clock on the Masscomp computer and on all of the consoles. The function call is ClockCommand(params). Table 2.9-27 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
params	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
time_desc	pointer to struct tm	Standard C Library.
cur_time	time_t	Standard C Library.
new_time	time_t	Standard C Library.
month	int	Standard C type.
day	int	Standard C type.
year	int	Standard C type.
hour	int	Standard C type.
min	int	Standard C type.
args_read	short	Standard C type.
date_given	short	Standard C type.
i	short	Standard C type.
Calls		
Function	**Where Described**	
LocalToGMTTime	Section 2.9.2.1.2.	
MCC_SystemError	Section 2.21.1.11.3.	
SendMCCMsg0	Macro defined in /simnet/mcc/include/bridge.h.	
Called By		
Function	**Where Described**	
cmd_clock	Section 2.9.1.2.2.	

Table 2.9-27: ClockCommand Information.

2.9.2.1.2 LocalToGMTTime

LocalToGMTTime converts the local time, *time*, into GMT time. The function call is LocalToGMTTime(time). Table 2.9-28 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
time	long	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ltime	pointer to struct tm	Standard C Library.
timezone	extern long	Standard C type.
daylight	extern int	Standard C type.
dst	long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
(time + timezone-dst)	long	GMT time equivalent of local time.
Called By		
Function	**Where Described**	
ClockCommand	Section 2.9.2.1.1.	

Table 2.9-28: LocalToGMTTime Information.

2.9.3 InterProcess Communication Status Command

The status of all ipc facilities used by the MCC may be reported using this command. It displays the status of semaphores used by internal message passing routines, message queues, parameter tables, vehicle simulator status tables, and AppleTalk packet information tables. Information such as processes attached to shared sections and outstanding messages are available. There is one CSU, ipc.c.

2.9.3.1 ipc.c

/simnet/mcc/Terminal/ipc.c

ipc.c contains routines for checking the status of the interprocess communication facilities used by the MCC. Table 2.9-29 describes the variables used by ipc.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
localtime()	extern pointer to struct tm	Standard C Library function.

Table 2.9-29: ipc.c Variable Information.

2.9.3.1.1 ReportIPCStatus

ReportIPCStatus reports the status of all the ipc facilities used by the MCC. The function call is ReportIPCStatus(). Table 2.9-30 describes the functions called using this function.

Calls	
Function	**Where Described**
DisplaySemStatus	Section 2.9.3.1.3.
DisplayShmStatus	Section 2.9.3.1.2.
DisplayMsgQStatus	Section 2.9.3.1.4.

Table 2.9-30: ReportIPCStatus Information.

2.9.3.1.2 DisplayShmStatus

DisplayShmStatus displays the shared memory status for various facilities used by the MCC, including the YUMM, Vehicle Table, Parameter Table and AppleTalk Info. *key* determines which shared memory will have its status displayed. The function call is DisplayShmStatus(key). Table 2.9-31 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
key	key_t	/usr/include/sys/types.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
shmid	int	Standard C type.
buf	struct shmid_ds	
lastProcess	int	Standard C type.
status	int	Standard C type.
time	pointer to struct tm	Standard C Library.
Calls		
Function	**Where Described**	
MCC_SystemError	Section 2.21.1.11.3.	
LookupProcessIdentifier	Section 2.21.1.26.2.	
Called By		
Function	**Where Described**	
ReportIPCStatus	Section 2.9.3.1.1.	
Called By		
Function	**Where Described**	
cmd_status_all	Section 2.9.1.2.10.	
cmd_status_ipc	Section 2.9.1.2.12.	

Table 2.9-31: DisplayShmStatus Information.

2.9.3.1.3 DisplaySemStatus

DisplaySemStatus is used to display the YUMM and MCC Semaphore status. *key* determines which status will be displayed. The function call is DisplaySemStatus(key, num). Table 2.9-32 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
key	key_t	/usr/include/sys/types.h
num	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
semid	int	Standard C type.
semnum	int	Standard C type.
buf	struct semid_ds	/usr/include/sys/sem.h
lastOp	int	Standard C type.
lastMCCOp	int	Standard C type.
status	int	Standard C type.
cur_count	int	Standard C type.
value	int	Standard C type.
zero_count	int	Standard C type.
otime	pointer to struct tm	Standard C Library.
Calls		
Function	**Where Described**	
MCC_SystemError	Section 2.21.1.11.3.	
LookupProcessIdentifier	Section 2.21.1.26.2.	
Called By		
Function	**Where Described**	
ReportIPCStatus	Section 2.9.3.1.1.	

Table 2.9-32: DisplaySemStatus Information.

2.9.3.1.4 DisplayMsgQStatus

DisplayMsgQStatus displays the YUMM Message queue status. The function call is DisplayMsgQStatus(key). Table 2.9-33 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
key	key_t	/usr/include/sys/types.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
msgid	int	Standard C type.
buf	struct msqid_ds	/usr/include/sys/msg.h
rtime	pointer to struct tm	Standard C Library.
stime	pointer to struct tm	Standard C Library.
lastDent	int	Standard C type.
lastRcvd	int	Standard C type.
creator	int	Standard C type.
Calls		
Function	**Where Described**	
MCC_SystemError	Section 2.21.1.11.3.	
LookupProcessIdentifier	Section 2.21.1.26.2.	

Called By	
Function	**Where Described**
ReportIPCStatus	Section 2.9.3.1.1.

Table 2.9-33: DisplayMsgQStatus Information.

2.9.4 Performance Monitor Command

The amount of shared memory used by several internal tables may be monitored using this command. This is accomplished by one CSU, perf.c.

2.9.4.1 perf.c

/simnet/mcc/Terminal/perf.c

perf.c contains a function which reports various performance and memory statistics.

2.9.4.1.1 ReportPerformance

ReportPerformance reports on the number of packets read and missed. It sends out a warning if too many packets have been missed. It also reports the size of the Vehicle List, Vehicle Status and Activation List memories. The function call is ReportPerformance(cardMemory). Table 2.9-34 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
cardMemory	pointer to Card_Memory_t	/simnet/mcc/Terminal/perf.c
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
Missed	long	Standard C type.
Pct	float	Standard C type.
Called By		
Function	**Where Described**	
cmd_perf	Section 2.9.1.2.6.	

Table 2.9-34: ReportPerformance Information.

2.9.5 Process Monitor Command

The state of various MCC processes may be displayed using this command. Each process may be in one of three states:

- Unspawned,
- Alive and well, or
- Dead.

This command can be used to identify processes which have aborted due to some unknown or unexpected condition. There is one CSU, process.c.

2.9.5.1 process.c
/simnet/mcc/Terminal/process.c

process.c contains a function to determine the current state of an MCC process. Table 2.9-35 describes the variables used by process.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
states	pointer to array 3 of char	Standard C type.

Table 2.9-35: process.c Variable Information.

2.9.5.1.1 ProcessStatus

ProcessStatus determines the status of a process. The function call is ProcessStatus(). Table 2.9-36 describes the internal variables used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
processStatus	int	Standard C type.
myPID	int	Standard C type.
Called By		
Function	**Where Described**	
cmd_status_all	Section 2.9.1.2.10.	
cmd_status_process	Section 2.9.1.2.14.	

Table 2.9-36: ProcessStatus Information.

2.9.6 Status Report Command

The Status Report Command handles parsing of requested status information. A request for status of ipc facilities or MCC processes causes a dispatch to the ipc or process module respectively. All other status reports are handled internally in this module. Currently, status information can be queried about:

- Interprocess Communication,
- Network Adapter Unit,
- MCC Processes,
- Vehicle Simulators participating in an exercise, and
- YUMM internals (message passing facility).

Vehicle simulator status can be requested by trailer-trailer element bumper number or vehicle ID of the simulator and displays information such as simulator status, location, and appearance.

Network Adapter Unit statistics call the network interface library (libnetif) to display information about packets received, sent, lost, and other adapter specific capabilities such as carrier loss, collision detection, and corrupted packets. This second level CSC contains one CSU, status.c.

2.9.6.1 status.c
/simnet/mcc/Terminal/status.c

status.c contains routines to determine the status of various MCC processes. It is called through the MCC host's console terminal. Table 2.9-37 describes the variables used by status.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
Veh_Name_Map	Name_Map	/simnet/mcc/Terminal/status.c

Table 2.9-37: status.c Variable Information.

2.9.6.1.1 SimStatusReport

SimStatusReport reports on the status of a specified simulator, *sim*. The function call is SimStatusReport(sim). Table 2.9-38 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	pointe to VehicleIDStatus	/simnet/mcc/include/veh_table.h
VAPDU	pointer to VehicleAppearanceVariant	pro_sim.h
location	TDB_POINT	/simnet/common/libsrc/libtdb/tdb.h
str	array 20 of char	Standard C type.
i	int	Standard C type.
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
xy_to_utm		
Called By		
Function	**Where Described**	
VehicleStatusReport	Section 2.9.6.1.2.	
cmd_status_all	Section 2.9.1.2.10.	
cmd_status_bumper	Section 2.9.1.2.11.	
cmd_status_sim	Section 2.9.1.2.15.	
cmd_status_sims	Section 2.9.1.2.16.	

Table 2.9-38: SimStatusReport Information.

2.9.6.1.2 VehicleStatusReport

VehicleStatusReport reports on the status of a particular vehicle ID, *vehicleID*. The function call is VehicleStatusReport(vehicleID). Table 2.9-39 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicleID	pointer to VehicleID	/simnet/common/include/protocol/basic.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
veh	register pointer to VehicleIDStatus	/simnet/mcc/include/veh_table.h
VAPDU	pointer to VehicleAppearanceVariant	/simnet/common/include/protocol/p_sim.h
useString	pointer to array of char	Standard C type.
stateString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
FQueue_Retrieve	Section 2.21.1.13.7.	
SimStatusReport	Section 2.9.6.1.1.	
cmd_status_vid	Section 2.9.1.2.17.	

Table 2.9-39: VehicleStatusReport Information.

2.9.7 Simulator Save/Restore

The Simulator Save/Restore commands allow the operator to save the internal MCC simulator tables to a disk file so that they may later be restored. This allows an MCC to capture the state of all of its manned simulators so that they can easily be restored at a later time from this "snapshot." The operator may save and restore simulator tables by specifying a scenario name. Any number of scenarios may be stored depending only on the amount of disk storage space available. An operator may choose from any one of the previously saved scenarios to restore. Upon restoring an exercise, all simulators are automatically initialized at their saved position and status, and the information is propagated to all of the appropriate Macintosh consoles. There is one CSU, vehicle.c.

2.9.7.1 vehicle.c

/simnet/mcc/Terminal/vehicle.c

vehicle.c contains routines for reporting the status of vehicle simulators. Table 2.9-40 describes the variable used by vehicle.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
tdb_errno	extern int	Standard C type.

Table 2.9-40: vehicle.c Variable Information.

2.9.7.1.1 SimListSaves

SimListSaves lists all files in the simulator save directory. The function call is SimListSaves(params). Table 2.9-41 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
params	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
command	array 500 of char	Standard C type.
Called By		
Function	**Where Described**	
cmd_list	Section 2.9.1.2.5.	

Table 2.9-41: SimListSaves Information.

2.9.7.1.2 SimSaveAll

SimSaveAll saves the simulator states into a specified file, *file*, placed in the saveDirectory. The function call is SimSaveAll(file). Table 2.9-42 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
file	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
saveFile	array MAXPATHLEN of char	Standard C type.
fd	int	Standard C type.
i	register short	Standard C type.
sim	register pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
veh	pointer to VehicleIDStatus	/simnet/mcc/inclujde/veh_table.h
astAppearance	pointer to VehicleAppearanceVariant	pro_sim.h
mbuf	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
rollPitchYaw	array 3 of float	Standard C type.
utmLocation	array 2TDB_PREC+3 of char	Standard C type.
location	TDB_POINT	/simnet/common/libsrc/libtdb/tdb.h

Calls	
Function	**Where Described**
FQueue_Retrieve	Section 2.21.1.13.7.
DecodeHullToWorldMatrix	Section 2.21.1.10.1.
rad_to_mil	Macro defined in /simnet/common/include/global//sim_macros.h.
xy_to_utm	
cmd_save	Section 2.9.1.2.9.

Table 2.9-42: SimSaveAll Information.

2.9.7.1.3 SimRestoreAll

SimRestoreAll restores simulators from a specified file, *file*. The function call is SimRestoreAll(file). Table 2.9-43 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
file	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
saveFile	array MAXPATHLEN of char	Standard C type.
fd	int	Standard C type.
buffer	array MSG_SIZE of char	Standard C type.
msg	pointer to MCCMessageBuffer	/simnet/mcc/include/MCC_ipc.h
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
vehicle	int	Standard C type.
company	unsigned char	Standard C type.
rspKind	long	Standard C type.
errCode	int	Standard C type.
rspLen	int	Standard C type.
utmLocation	array 2TDB_PREC+3 of char	Standard C type.
location	TDB_POINT	/simnet/common/libsrc/libtdb/tdb.h
Calls		
Function	**Where Described**	
xy_to_utm		
Transact	Section 2.21.2.10.9.	
SendMCCMsg1	Macro defined in /simnet/mcc/include/bridge.h.	
Called By		
Function	**Where Described**	
cmd_restore	Section 2.9.1.2.8.	

Table 2.9-43: SimRestoreAll Information.

2.9.7.1.4 VehFromSimNum

VehFromSimNum looks up a simulator by its simulator number, given by *str*. Valid simulator numbers consist of a <number><letter> string pair, NOT separated by any spaces. VehFromSimNum returns a pointer to the SimulatorStatus object describing the simulator, or 0 if none is found with that number. The function call is VehFromSimNum(str). Table 2.9-44 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
str	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
i	register short	Standard C type.
trailer	int	Standard C type.
trailerElement	char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	pointer to SimulatorStatus	No simulator found with that number.
sim	pointer to SimulatorStatus	Simulator found.
Called By		
Function	**Where Described**	
cmd_status_sim	Section 2.9.1.2.15.	

Table 2.9-44: VehFromSimNum Information.

2.9.7.1.5 VehFromBumper

VehFromBumper looks up a simulator by its bumper number, given by *str*. Valid bumper numbers are <company><bumper #> pairs where the company is identified by the letter A, B, C, D, F, H, L or S. VehFromBumper returns a pointer to the SimulatorStatus object describing the simulator, or 0 if none is found with that number. The function call is VehFromBumper(str). Table 2.9-45 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
str	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	pointer to SimulatorStatus	/simnet/mcc/include/veh_table.h
i	register short	Standard C type.
bumper	int	Standard C type.
company	char	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
0	pointer to SimulatorStatus	No simulator found with that number.
sim	pointer to SimulatorStatus	Simulator found.
Called By		
Function	**Where Described**	
veh_status_bumper	Section 2.9.1.2.11.	

Table 2.9-45: VehFromBumper Information.

2.9.7.1.6 ValidSimNum

ValidSimNum determines if a string, *cp*, is a valid simulator number, i.e. one or two digits followed by a letter. The function call is ValidSimNum(cp). Table 2.9-46 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
cp	register pointer to char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Sim number not valid.
1	int	Sim number valid.

Table 2.9-46: ValidSimNum Information.

2.9.7.1.7 ValidBumperNumber

ValidBumperNumber determines if a string, *cp*, is a valid bumper number, i.e. a letter followed by one or two digits. The function call is ValidBumperNumber(cp). Table 2.9-47 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
cp	register pointer to char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Bumper number not valid.
1	int	Bumper number valid.

Table 2.9-47: ValidBumperNumber Information.

2.10 Masscomp Communication Software

This CSC allows communication between the process on the Masscomp® 5600 and the Bridge console. The ATSend and ATRecv processes correctly the appropriate encoding and decoding of messages that are sent over the RS-232 cable connecting the two computers. The structure of the CSC is shown in Figure 2.10-1.

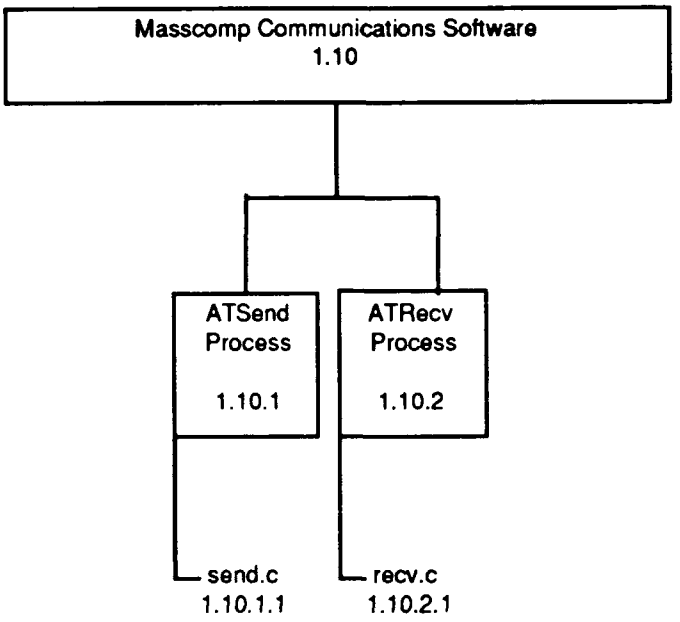


Figure 2.10-1: Masscomp Communication Software Structure.

The CSC has two second level CSCs associated with it:

ATSend Process
ATRecv Process

2.10.1 ATSend Process

The MCC AppleTalk Send Process is responsible for sending messages from the MCC host system to one or more Apple Macintosh consoles. The ATSend process receives messages from MCC host processes and performs the necessary encapsulation required by the AppleTalk Bridge. The encapsulated messages are transmitted across a low speed (9600 baud) serial line to an AppleTalk Bridge. The Bridge acts as a physical layer router between the AppleTalk Network and the serial line connected to the MCC host system.

The encapsulation performed includes:

- Assembling a short header which specifies the originator of the message,
- Prefixing a header for packet framing information, and
- Adding a checksum to the packet.

An interframe gap of 100 milliseconds allows the AppleTalk Bridge enough time to process and transmit the packet onto the AppleTalk network, obviating the need for more complex flow control between the MCC host and Bridge system. One CSU realizes this functionality, send.c.

2.10.1.1 send.c

/simnet/mcc/ATSend/send.c

send.c contains routines which implement the AppleTalk network send process. Table 2.10-1 describes the variables used by send.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
fd	int	Standard C type.
hexDigit	array of char	Standard C type.

Table 2.10-1: send.c Variable Information.

2.10.1.1.1 main

main is the entry point to the ATSend process. The function call is main(argc, argv). Table 2.10-2 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tio	struct termio	/usr/include/termio.h
Calls		
Function	**Where Described**	
InitializesProcess	Section 2.21.1.19.1.	
MCC_SystemError	Section 2.21.1.11.3.	
Unlock	Section 2.21.2.6.4.	

Table 2.10-2: main Information.

2.10.1.1.2 ProcessMessage

ProcessMessage handles a message requesting a DDP packet to be sent. *type* indicates whether the message is a one-way, asynchronous or sychronous request, or a response to a previous request. *kind* indicates the type of message which has been received. *length* indicates the length of the data in the message received. *data* is a pointer to the actual message data. Information is extracted from the message data based on the *kind* of message. The function call is ProcessMessage(type, kind, length, data). Table 2.10-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
kind	long	Standard C type.
length	int	Standard C type.
data	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
len	int	Standard C type.
i	register int	Standard C type.
cp	register pointer to char	Standard C type.
bp	register pointer to char	Standard C type.
buf	array of bridgeFrameSize char	Standard C type.
Calls		
Function	**Where Described**	
PutByte	Macro defined in /simnet/mcc/ATSend/send.c.	
MCC_SystemError	Section 2.21.1.11.3.	

Table 2.10-3: ProcessMessage Information.

2.10.2 ATRecv Process

The AppleTalk(tm) Receive Process is responsible for processing AppleTalk Network packets passed to the MCC host over a serial line connection. Typically, an Apple Macintosh computer is used as a physical layer router between the AppleTalk Network and the serial line used by the MCC host system to communicate with the MCC Macintosh consoles.

ATRecv reads in each packet passed over the serial line, performs error checksumming, decodes the message type in the packet, and dispatches the message to the specified recipient. There is one CSU, recv.c.

2.10.2.1 recv.c
/simnet/mcc/ATRecv/recv.c

recv.c contains routines which implement the AppleTalk network receive process.

2.10.1.2.1 main

main is the entry point to the ATRecv process. The function call is main(argc, argv). Table 2.10-4 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
argc	int	Standard C type.
argv	pointer to pointer to char	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
fd	int	Standard C type.
len	int	Standard C type.
fmLen	int	Standard C type.
fmChk	int	Standard C type.
fmSrc	int	Standard C type.
fmDst	int	Standard C type.
i	register int	Standard C type.
cp	register pointer to char	Standard C type.
bp	register pointer to char	Standard C type.
tio	struct termio	/usr/include/termio.h
client	YUMMSocket	/simnet/rcs/libipc/yumm.h
msg	MCCMessageBuffer	/simnet/mcc/include/MCC_ipc .h
buf	array bridgeFrameSize of char	Standard C type.
Errors		
Error Name	**Reason for Error**	
MCC_BAD_FRAME	Bad frame received from bridge.	
MCC_DDP_OVERRUN	No buffer for incoming DDP datagram.	
Calls		
Function	**Where Described**	
InitializeProcess	Section 2.21.1.19.1.	
MCC_SystemError	Section 2.21.1.11.3.	
GetByte	Macro defined in /simnet/mcc/ATRecv/recv.c.	
ATAllocBuffer	Section 2.21.1.2.1.	
ATFreeBuffer	Section 2.21.1.2.2.	
SendMsg	Section 2.21.2.10.6.	

Table 2.10-4: main Information.

2.11 The Bridge Console

The Bridge Console is a Macintosh application that relays information between an AppleTalk network and an RS-232 serial line. This relay operates above the link layer. AppleTalk LAP frames are received by Bridge and retransmitted on the serial line in encoded form. Encoded frames are received on the serial line and retransmitted as AppleTalk LAP frames.

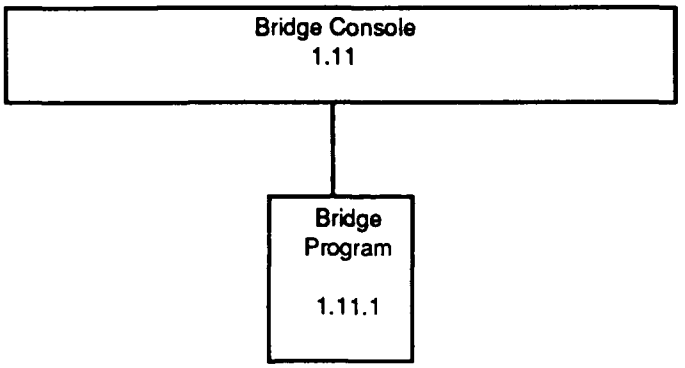


Figure 2.11-1: Bridge Console Structure.

2.11.1 Bridge Program

2.11.1.1 HexDigit

HexDigit converts *i* to hexidecimal digits. The function call is HexDigit(i:INTEGER):INTEGER. Table 2.11-1 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
j	INTEGER	Standard Pascal Type.
Called By		
Function	**Where Described**	
EncodeFrame	Section 2.11.1.6.	
ValidFrame	Section 2.11.1.7.	
SendAnswer	Section 2.11.1.10.	

Table 2.11-1: HexDigit Information.

2.11.1.2 BinNibble

BinNibble nibbles *i* into 4 byte binary chunks. The function call is BinNibble(i:INTEGER):INTEGER. Table 2.11-2 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Called By		
Function	**Where Described**	
ValidFrame	Section 2.11.1.7.	
DecodeFrame	Section 2.11.1.8.	

Table 2.11-2: BinNibble Information.

2.11.1.3 QueueARead

QueueARead reads a packet off the Appletalk and adds it to the read queue. The procedure call is QueueARead(i:INTEGER). Table 2.11-3 describes the parameters used, errors returned and functions called using this procedure.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
errCode	INTEGER	Standard Pascal Type.
Errors		
Error Name	**Reason for Error**	
Error while trying to queue an AppleTalk read.	Call to LAPRead failed.	
Calls		
Function	**Where Described**	
LAPRead	Standard Appletalk Manager function for Macintosh.	
Called By		
Function	**Where Described**	
ChkSerOut	Section 2.11.1.15.	
SetUp	Section 2.11.1.16.	
Bridge	Section 2.11.1.17.	

Table 2.11-3: QueueARead Information.

2.11.1.4 QueueAWrite

QueueAWrite writes a packet to the Appletalk network from the write queue. The procedure call is QueueAWrite(i:INTEGER). Table 2.11-4 describes the parameters used, errors returned and functions called using this procedure.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
errCode	INTEGER	Standard Pascal Type.
Errors		
Error Name	**Reason for Error**	
Error while tryig to queue an AppleTalk write.	Call to LAPWrite failed.	
Calls		
Function	**Where Described**	
LAPWrite	Standard Appletalk Manager function for Macintosh.	
Called By		
Function	**Where Described**	
EndOfFrame	Section 2.11.1.13.	

Table 2.11-4: QueueAWrite Information.

2.11.1.5 QueueSWrite

QueueSWrite writes a packet from the queue to the serial line connected to the MCC host. The procedure call is QueueSWrite(i:INTEGER). Table 2.11-5 describes the parameters used, errors returned and functions called using this procedure.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
errCode	INTEGER	Standard Pascal Type.
Errors		
Error Name	**Reason for Error**	
Error while trying to queue a serial write.	Call to PBWrite failed.	
Calls		
Function	**Where Described**	
PBWrite	Standard Device Manager function for Macintosh.	

Called By	
Function	**Where Described**
Bridge	Section 2.11.1.17.

Table 2.11-5: QueueSWrite Information.

2.11.1.6 EncodeFrame

EncodeFrame fills in fields of a packet according to standard Appletalk format. The procedure call is EncodeFrame(i:INTEGER). Table 2.11-6 describes the parameters used and functions called using this procedure.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
len	INTEGER	Standard Pascal Type.
j	INTEGER	Standard Pascal Type.
k	INTEGER	Standard Pascal Type.
ch	INTEGER	Standard Pascal Type.
Calls		
Function	**Where Described**	
HexDigit	Section 2.11.1.1.	
Called By		
Function	**Where Described**	
Bridge	Section 2.11.1.17.	

Table 2.11-6: EncodeFrame Information.

2.11.1.7 ValidFrame

ValidFrame checks a packet to see if its checksum matches its date. The function call is ValidFrame(i:INTEGER):BOOLEAN. Table 2.11-7 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
len	INTEGER	Standard Pascal Type.
j	INTEGER	Standard Pascal Type.
k	INTEGER	Standard Pascal Type.

Calls	
Function	**Where Described**
BinNibble	Section 2.11.1.2.
HexDigit	Section 2.11.1.1.
Called By	
Function	**Where Described**
EndOfFrame	Section 2.11.1.13.

Table 2.11-7: ValidFrame Information.

2.11.1.8 DecodeFrame

DecodeFrame takes a packet sent from the host and adds information to make it an Appletalk packet to send to one of the consoles. The procedure call is DecodeFrame(i:INTEGER). Table 2.11-8 describes the parameters used, errors returned and functions called using this procedure.

Parameters		
Parameter	**Type**	**Where Type Declared**
v	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
len	INTEGER	Standard Pascal Type.
j	INTEGER	Standard Pascal Type.
k	INTEGER	Standard Pascal Type.
ch	INTEGER	Standard Pascal Type.
Calls		
Function	**Where Described**	
BinNibble	Section 2.11.1.2.	
Called By		
Function	**Where Described**	
EndOfFrame	Section 2.11.1.13.	

Table 2.11-8: DecodeFrame Information.

2.11.1.9 QueryFrame

QueryFrame determines if this frame is a query frame which is asking for the address of the bridge. The function call is QueryFrame(i:INTEGER):BOOLEAN. Table 2.11-9 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Called By		
Function	**Where Described**	
EndOfFrame	Section 2.11.1.13.	

Table 2.11-9: QueryFrame Information.

2.11.1.10 SendAnswer

SendAnswer sends the node and net address of the bridge back to the requester. The procedure call is SendAnswer. Table 2.11-10 describes the internal variables used, errors returned and functions called using this procedure.

Internal Variables		
Variable	**Type**	**Where Type Declared**
errcode	INTEGER	Standard Pascal Type.
myNode	INTEGER	Standard Pascal Type.
myNet	INTEGER	Standard Pascal Type.
len	LongInt	
buf	PACKED ARRAY [1...7] OF Byte	
Errors		
Error Name	**Reason for Error**	
Error in obtaining node address.	Call to GetNodeAddress failed.	
Error in sending query reply.	Call to FSWrite failed.	
Calls		
Function	**Where Described**	
GetNodeAddress	Standard Appletalk Manager function for Macintosh.	
HexDigit	Section 2.11.1.1.	
FSWrite	Standard Device Manager function for Macintosh.	
Called By		
Function	**Where Described**	
EndOfFrame	Section 2.11.1.13.	

Table 2.11-10: SendAnswer Information.

2.11.1.11 StartOfFrame

StartOfFrame finds a buffer available to store the new packet from the serial line. The procedure call is StartOfFrame. Table 2.11-11 describes the internal variables used by this procedure.

Internal Variables		
Variable	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Called By		
Function	**Where Described**	
ChkSerIn	Section 2.11.1.14.	

Table 2.11-11: StartOfFrame Information.

2.11.1.12 AppendByte

AppendByte appends the inputted character to the end of the current buffer. The procedure call is AppendByte(ch:INTEGER). Table 2.11-12 describes the parameters used by this procedure.

Parameters		
Parameter	**Type**	**Where Type Declared**
ch	INTEGER	Standard Pascal Type.
Internal Variables		
Variable	**Type**	**Where Type Declared**
i	INTEGER	Standard Pascal Type.
Called By		
Function	**Where Described**	
ChkSerIn	Section 2.11.1.14.	

Table 2.11-12: AppendByte Information.

2.11.1.13 EndOfFrame

EndOfFrame checks if frame is valid and if it is, queues it to be written to the Appletalk net. If current frame is a query frame, sends node and net address. The procedure call is EndOfFrame. Table 2.11-13 describes the errors returned and functions called using this procedure.

Errors	
Error Name	**Reason for Error**
Incorrect frame from serial line.	Frame type unknown.

Calls	
Function	**Where Described**
QueryFrame	Section 2.11.1.9.
SendAnswer	Section 2.11.1.10.
ValidFrame	Section 2.11.1.7.
DecodeFrame	Section 2.11.1.8.
QueueAWrite	Section 2.11.1.4.
Called By	
Function	**Where Described**
ChkSerIn	Section 2.11.1.14.

Table 2.11-13: EndOfFrame Information.

2.11.1.14 ChkSerIn

ChkSerIn checks to see if any characters have arrived on the serial line and loads them into a local buffer. The procedure call is ChkSerIn. Table 2.11-14 describes the internal variables used, errors returned and functions called using this procedure.

Internal Variables		
Variable	**Type**	**Where Type Declared**
errCode	INTEGER	Standard Pascal Type.
i	INTEGER	Standard Pascal Type.
ch	INTEGER	Standard Pascal Type.
arrived	LongInt	
str	*PACKED ARRAY [1..100] OF* Byte	
Errors		
Error Name	**Reason for Error**	
SerGetBuf failed.	Call to SerGetBuf failed.	
FSRead on serial port failed.	Call to FSRead failed.	
Calls		
Function	**Where Described**	
SerGetBuf	Standard Serial Drivers function for Macintosh.	
FSRead	Standard Device Manager function for Macintosh.	
EndOfFrame	Section 2.11.1.13.	
AppendByte	Section 2.11.1.12.	
StartOfFrame	Section 2.11.1.11.	
Called By		
Function	**Where Described**	
Bridge	Section 2.11.1.17.	

Table 2.11-14: ChkSerIn Information.

2.11.1.15 ChkSerOut

ChkSerOut looks for completed serial transmissions. The procedure call is ChkSerOut. Table 2.11-15 describes the internal variables used, errors returned and functions called using this procedure.

Internal Variables		
Variable	**Type**	**Where Type Declared**
b	INTEGER	Standard Pascal Type.
Errors		
Error Name	**Reason for Error**	
Serial write failed.		
Calls		
Function	**Where Described**	
QueueARead	Section 2.11.1.3.	
Called By		
Function	**Where Described**	
Bridge	Section 2.11.1.17.	

Table 2.11-15: ChkSerOut Information.

2.11.1.16 SetUp

SetUp initializes the environment for the bridge Mac. It initializes the Mac application environment, the queue data structure, the Appletalk driver, and the serial driver. The procedure call is SetUp. Table 2.11-16 describes the internal variables used, errors returned and functions called using this procedure.

Internal Variables		
Variable	**Type**	**Where Type Declared**
tRect	Rect	
errCode	INTEGER	Standard Pascal Type.
i	INTEGER	Standard Pascal Type.
Errors		
Error Name	**Reason for Error**	
Error in opening AppleTalk.	Call to MPPOpen failed.	
Error in disabling DDP.	Call to LAPCloseProtocol failed.	
Error in opening default protocol handler.	Call to LAPOpenProtocol failed.	
Error in opening serial output driver.	Call to OpenDriver failed.	
Error in resetting serial output driver.	Call to SerReset failed.	
Error in opening serial input driver.	Call to OpenDriver failed.	
Error in resetting serial input driver.	Call to SerReset failed.	

Calls	
Function	**Where Described**
WWInit	Standard
InitGraf	Standard Quickdraw function for Macintosh.
InitFonts	Standard Font Manager function for Macintosh.
FlushEvents	Standard Operating System Events Manager function for Macintosh.
InitWindows	Standard Window Manager function for Macintosh.
InitCursor	Standard Quickdraw function for Macintosh.
SetRect	Standard Quickdraw function for Macintosh.
WWNew	Standard
BufStrPtr	Standard
NewPtr	Standard Memory Manager function for Macintosh.
ABRecHandle	Standard Appletalk Manager function for Macintosh.
NewHandle	Standard Memory Manager function for Macintosh.
HLock	Standard Memory Manager function for Macintosh.
Handle	Standard Memory Manager function for Macintosh.
Ptr	Standard Memory Manager function for Macintosh.
ParmBlkPtr	Standard Device Manager function for Macintosh.
MPPOpen	Standard Appletalk Manager function for Macintosh.
LAPCloseProtocol	Standard Appletalk Manager function for Macintosh.
LAPOpenProtocol	Standard Appletalk Manager function for Macintosh.
OpenDriver	Standard Device Manager function for Macintosh.
SerReset	Standard Serial Drivers function for Macintosh.
QueueARead	Section 2.11.1.3.
Called By	
Function	**Where Described**
Bridge	Section 2.11.1.17.

Table 2.11-16: SetUp Information.

2.11.1.17 Bridge

Bridge is the main program of the Bridge Console. This program first calls SetUp in order to initialize the operating environment. Bridge then processes events. Information is relayed between the AppleTalk network and the RS-232 serial line. The program call is Bridge. Table 2.11-17 describes the internal variables used, errors returned and functions called using this program.

Internal Variables		
Variable	**Type**	**Where Type Declared**
myEvent	EventRecord	Standard Apple Type.
temp	BOOLEAN	Standard Pascal Type.
code	INTEGER	Standard Pascal Type.
whichWindow	WindowPtr	Standard Apple Type.
aRecvCount	INTEGER	Standard Pascal Type.
aXmitCount	INTEGER	Standard Pascal Type.
sRecvCount	INTEGER	Standard Pascal Type.
sXmitCount	INTEGER	Standard Pascal Type.
buffers	ARRAY [1..numberBuffers] of Buffer	Type defined in Bridge.
serialOut	INTEGER	Standard Pascal Type.
serialIn	INTEGER	Standard Pascal Type.
b	INTEGER	Standard Pascal Type.
availBuf	INTEGER	Standard Pascal Type.
traceMode	BOOLEAN	Standard Pascal Type.

Errors	
Error Name	**Reason for Error**
AppleTalk read failed.	
AppleTalk write failed.	

Calls	
Function	**Where Described**
SetUp	Section 2.11.1.16.
SystemTask	Standard Desk Manager function for Macintosh.
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.
FindWindow	Standard Window Manager function for Macintosh.
WWMouseDown	Standard
WindowPtr	Standard Window Manager function for Macintosh.
WWActivate	Standard
WWUpdateEvent	Standard
ABRecHandle	Standard Appletalk Manager function for Macintosh.
EncodeFrame	Section 2.11.1.6.
QueueSWrite	Section 2.11.1.5.
QueueARead	Section 2.11.1.3.
ChkSerIn	Section 2.11.1.14.
ChkSerOut	Section 2.11.1.15.

Table 2.11-17: Bridge Information.

2.12 Appletalk Network Software

Overview of the Network

The Management, Command, and Control System (MCC) uses the AppleTalk network to link all Macintosh computers with each other and to the MCC host system via a bridge link on the network to the host . AppleTalk is a proprietary network protocol developed and supported by Apple Computer Corporation to provide a vehicle for sharing information and transferring information from one Apple system to another in the network. This network protocol has been tied to the MCC host system via a programmed link on a Macintosh system which allows it to become the bridge from the AppleTalk network to the host system. In fact, it is called the Bridge Macintosh from Appletalk to the MCC host.

Appletalk Network Format

The AppleTalk network developed by Apple Computer involves not only a physical link for carrying data, but also a family of protocols for use over that link. These protocols are organized into a hierarchy of layers similar to that of the International Standardds Organization's (ISO) Open System Interconnection (OSI) basic reference model. Each protocol layer in the hierarchy provides services that are used, in turn, by the next higher layer to provide services that are more complex or specialized. The lower layers provide a means to group data into packets, addressing data packets to specific computers (nodes) on the network. The intermediate layers build on the lower layers to provide reliable data transfer, and provide a means for locating nodes on the network by name. The highest layers contain protocols for specific applications, such as controlling printers or accessing file servers.

Network Protocols

Within each computer (node) there are one or more logical entities, known as sockets. Nodes, and sockets within a node, are addressable from other nodes in the network by their unique name.

The MCC System directly uses the services of two AppleTalk protocols, called the Name Binding Protocol (NBP) and the AppleTalk Transaction Protocol (ATP). These protocols, in turn, use lower layer protocols called the Datagram Delivery Protocol (DDP), and the AppleTalk Link Access Protocol (ALAP). These four layers together with the AppleTalk cable provide the physical layer, data link layer, network layer, and transport layer for the network.

MCC/Macintosh to Host Implementation

The MCC host participates in the AppleTalk network through a Macintosh computer programmed to become the Bridge to the host. An RS-232 line runs between the host and the Bridge. The Macintosh application computers are tied to the Bridge Macintosh via an AppleTalk twisted pair shielded cable. Signalling on the cable is at 230.4 Kbaud, using standard RS-422 drivers and receivers.

A Macintosh application that is expecting to receive ATP requests sets up buffers that will be used to contain those those requestes when they arrive. The application is notified when a request arrive by an event it encounters in its main event program loop. Similarly, an application that is expecting to receive a response to a request it is issuing will supply a buffer to contain that response. An event wil signal the arrival of the response. All of the consoles (the application Macintoshes) make use of this feature, which allows them to engage the MCC host in ATP transactions while carrying on with oth er processing requirements.

The MCC host participates in the Appletalk network through the Bridge by host processes called ATSend and ATRecv. There is a shared memory area on the network called the AppleTalk Info shared memory area on the host, and a library of routines incorporated into each host console process. This shared memory area works through the ATRecv and ATSend processes and the host message system to send and receive messages from the host to the Bridge through to the AppleTalk network and back.

2.13 The SCC Console

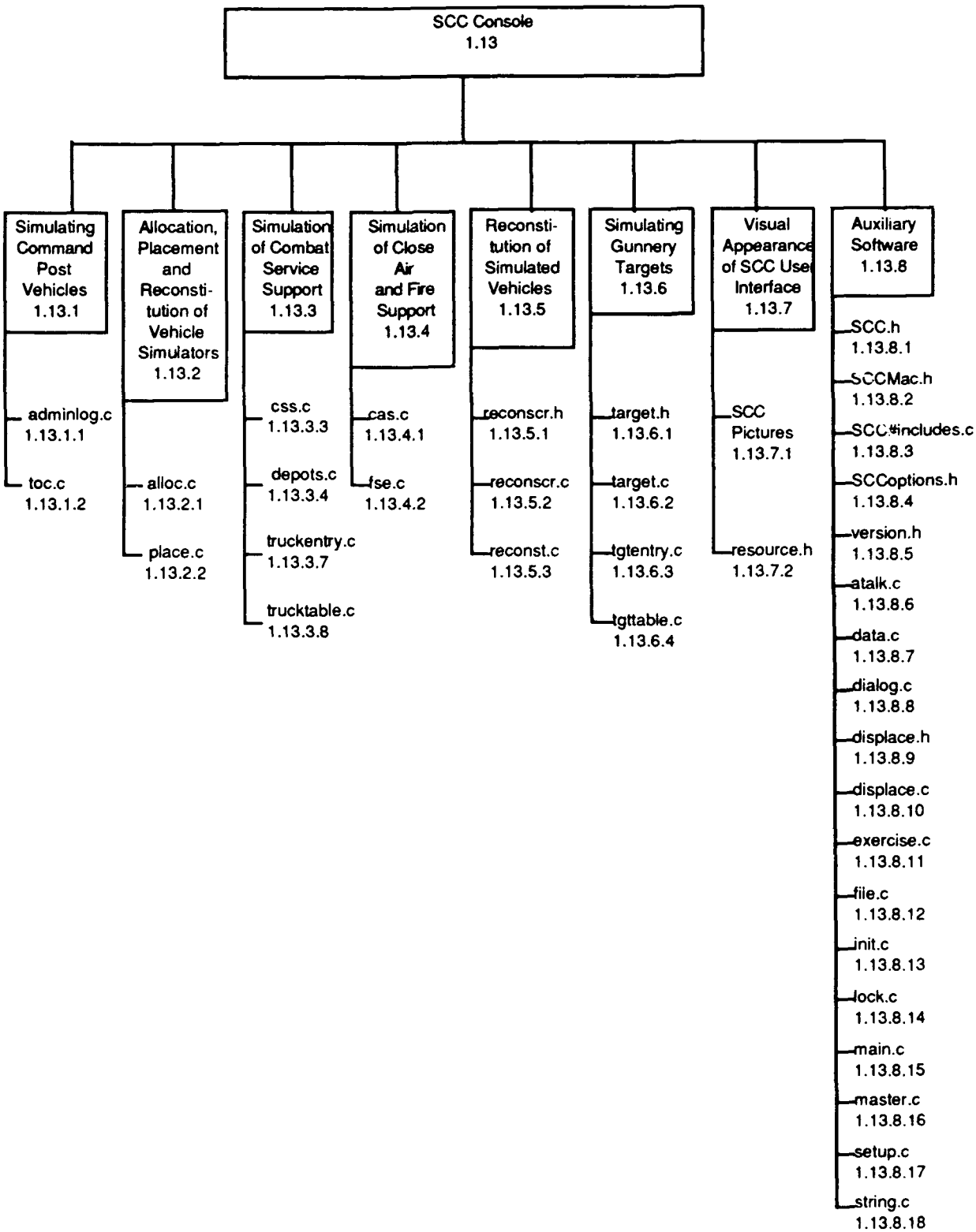


Figure 2.13-1: SCC Console Structure.

2.13.1 Simulating Command Post Vehicles

2.13.1.1 adminlog.c
Development:SIMNET:MCC:SCC:adminlog.c

adminlog.c contains routines for positioning and reconstituting the simulated Admin/Log Center vehicle. Table 2.13-1 describes the variables used by adminlog.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
alocRole	char	Standard C type.
alocLocation	MapCoordinates	Development:SIMNET:libmac:map.h
alocCancelField	PushButtonFieldDefn	Development:SIMNET:libmac:dialog.h
alocRBFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
alocLocationField	CoordinatesFieldDefn	Development:SIMNET:libmac:dialog.h
alocFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
initALOCDialog	DialogNodeDefn	Development:SIMNET:libmac:sequence.h
reconstALOCDialog	DialogNodeDefn	Development:SIMNET:libmac:sequence.h

Table 2.13-1: adminlog.c Variable *Information.*

2.13.1.1.1 LoadCannedALOC

LoadCannedALOC loads canned Admin/Log center initialization data. The function call is LoadCannedALOC(). Table 2.13-2 describes the function called using this function.

Calls	
Function	**Where Described**
StringToMapCoordinates	See Section 2.22.1.26.1.
Called By	
Function	**Where Described**
LoadCannedData	See Section 2.13.8.15.2.

Table 2.13-2: LoadCannedALOC Information.

2.13.1.1.2 UploadALOCParameters

UploadALOCParameters uploads information to the host about Admin/Log Center parameters. This function will only do something if VERSION is APPLETALK. Otherwise, it is just a dummy function.The function call is UploadALOCParameters(). Table 2.13-3 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCALOCRequest	Development:SIMNET:MCC:SCC:SCC.h
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
ALOCEvent	See Section 2.13.1.1.5.	

Table 2.13-3: UploadALOCParameters Information.

2.13.1.1.3 ALOCInitFetch

ALOCInitFetch labels the radio buttons that display the choice of forces in the Admin/Log Center Init dialog. The function call is ALOCInitFetch(state). Table 2.13-4 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
LabelForceButtons	See Section 2.22.1.16.1.	

Table 2.13-4: ALOCInitFetch Information.

2.13.1.1.4 ALOCReconstFetch

ALOCReconstFetch labels the radio buttons that display the choice of forces in the Admin/Log Center Reconst dialog. The function call is ALOCReconstFetch(state). Table 2.13-5 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
LabelForceButtons	See Section 2.22.1.16.1.	

Table 2.13-5: ALOCReconstFetch Information.

2.13.1.1.5 ALOCEvent

ALOCEvent is called when an Admin/Log Center Initialize or Reconstitute Dialog button is pressed. If the OK button is pressed, the validity of the data is checked, the ALOC parameters are uploaded, and an entry is made in the table of displaceable vehicles. If the Cancel button is pressed, the current dialog is taken down and the overview dialog is put up. The function call is ALOCEvent(state, itemNo). Table 2.13-6 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Do nothing upon return.
okBItem	int	Resource id of OK button.
itemNo	int	Resource id of button hit.
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
UploadALOCParameters	See Section 2.13.1.1.2.	

Table 2.13-6: ALOCEvent Information.

2.13.1.2 toc.c

Development:SIMNET:MCC:SCC:toc.c

toc.c contains routines for positioning and reconstituting the simulated Tactical Operations Center vehicles. Table 2.13-7 describes the variables used by toc.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tocRole	char	Standard C type.
tocLocation	MapCoordinates	Development:SIMNET:libmac: map.h
tocConfiguration	char	Standard C type.
tocCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
tocRBFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
tocLocationField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
tocFieldList	array of FieldDefn	Development:SIMNET:libmac: dialog.h
initTOCDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
reconstTOCDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-7: toc.c Variable Information.

2.13.1.2.1 LoadCannedTOC

LoadCannedTOC loads Tactical Operations Center canned data. The function call is LoadCannedTOC(). Table 2.13-8 describes the function called using this function.

Calls	
Function	**Where Described**
StringToMapCoordinates	See 2.22.1.26.1.
Called By	
Function	**Where Described**
LoadCannedData	See Section 2.13.8.15.2.

Table 2.13-8: LoadCannedTOC Information.

2.13.1.2.2 UploadTOCParameters

UploadTOCParameters uploads Tactical Operations Center parameters to the host. This function will only do something if VERSION is APPLETALK. Otherwise, it is just a dummy function.The function call is UploadTOCParameters(). Table 2.13-9 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCTOCRequest	Development:SIMNET:MCC: SCC:SCC.h
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	

Called By	
Function	**Where Described**
TOCEvent	See Section 2.13.1.2.5.

Table 2.13-9: UploadTOCParameters Information.

2.13.1.2.3 TOCInitFetch

TOCInitFetch labels the radio buttons that display the choice of forces in the Tactical Operations Center Init dialog. The function call is TOCInitFetch(dialog). Table 2.13-10 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
LabelForceButtons	See Section 2.22.1.16.1.	

Table 2.13-10: TOCInitFetch Information.

2.13.1.2.4 TOCReconstFetch

TOCReconstFetch labels the radio buttons that display the choice of forces in the Tactical Operations Center Reconst dialog. The function call is TOCReconstFetch(dialog). Table 2.13-11 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
LabelForceButtons	See Section 2.22.1.16.1.	

Table 2.13-11: TOCReconstFetch Information.

2.13.1.2.5 TOCEvent

TOCEvent is called when a button in the Tactical Operations Center dialog is pressed. If the OK button is pressed, the validity of the data is checked, the TOC parameters are uploaded, and an entry is made in the table of displaceable vehicles. If the Cancel button is pressed, the current dialog is taken down and the overview dialog is put up. The function call is TOCEvent(dialog, itemNo). Table 2.13-12 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
okBItem	int	resource id of the OK button
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
UploadTOCParameters	See Section 2.13.1.2.2.	

Table 2.13-12: TOCEvent Information.

2.13.2 Allocation, Placement and Reconstitution of Vehicle Simulators

2.13.2.1 alloc.c

Development:SIMNET:MCC:SCC:alloc.c

alloc.c implements the user interface for allocating vehicle simulators to organizational units. Table 2.13-13 describes the variables used by alloc.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
allocTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
allocColumns	extern array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
allocEntryDialog	extern DialogDefn	Development:SIMNET:libmac: dialog.h
allocatedSim	pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
allocColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
allocTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
allocTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
allocTableDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
allocEntryFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
allocEntryFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
allocEntryDialog	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.13-13: alloc.c Variable Information.

2.13.2.1.1 SetUpAlloc

SetUpAlloc sets up the Simulator Allocate Table dialog for allocating vehicles. This is the table of simulators currently available for allocation to units. The function call is SetUpAlloc(). Table 2.13-14 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
Called By	
Function	**Where Described**
SetUp	See Section 2.13.8.17.1.

Table 2.13-14: SetUpAlloc Information.

2.13.2.1.2 AllocTableFetch

AllocTableFetch reads the current values of the present table of simulators available for allocation to units into the dialog fields to display to the user. The function call is AllocTableFetch(dialog). Table 2.13-15 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
SimTableSetup	See Section 2.22.2.10.1.	

Table 2.13-15: AllocTableFetch Information.

2.13.2.1.3 AllocTableSelect

AllocTableSelect is called when an entry in the allocation table is selected. It checks for validity of the selection and brings up the next dialog describing the selection. The function call is AllocTableSelect(defn, row, box, event). Table 2.13-16 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
ScrollTableRowToEntry	See Section 2.22.1.34.1.	
SetScrollTableSelection	See Section 2.22.1.37.2.	
SimTableEntry	See Section 2.22.2.10.3.	
ShowCaution	See Section 2.22.1.4.1.	
ShowDialog	See Section 2.22.1.11.1.	
ZoomUpFromTableEntry	See Section 2.13.8.8.2.	

Table 2.13-16: AllocTableSelect Information.

2.13.2.1.4 AllocTableEvent

AllocTableEvent handles an event in an Allocation Table dialog. If the Overview button was pressed, the Scroll Table is discarded and the overview screen is put up. The function call is AllocTableEvent(dialog, itemNo). Table 2.13-17 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button pressed
Calls		
Function	**Where Described**	
DisposeScrollTable	See Section 2.22.1.30.2.	

Table 2.13-17: AllocTableEvent Information.

2.13.2.1.5 AllocEntryFetch

AllocEntryFetch reads the current values of the simulator allocation entry into the dialog fields to display to the user. The simulator and type number fields are set. The radio buttons are set. Radio buttons for units not participating in the exercise are disabled. The function call is AllocEntryFetch(dialog). Table 2.13-18 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 20 char	Standard C type.
pStr	pointer to char	Standard C type.
i	short	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
SimulatorTypeCStr	See Section 2.22.2.9.2.	
DisableControl	See Section 2.22.1.7.1.	

Table 2.13-18: AllocEntryFetch Information.

2.13.2.1.6 AllocEntryEvent

AllocEntryEvent is called when a button in the Simulator Allocate Entry dialog is pressed. If the OK button is pressed, the simulator entry is allocated to the company, the scroll entry table is taken down, and the overview screen is put up. If the Help button is pressed, the Help dialog is put up. The function call is AllocEntryEvent(dialog, itemNo). Table 2.13-19 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SimAllocRequest	Development:SIMNET:MCC:include:sim_xact.h
Calls		
Function	**Where Described**	
showhelp	See Section 2.22.1.19.4.	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	
UpdateScrollTableEntry	See Section 2.22.1.31.1.	
ZoomDownToTableEntry	See Section 2.13.8.8.3.	
ThrowDialog	See Section 2.22.1.11.3.	

Table 2.13-19: AllocEntryEvent Information.

2.13.2.2 place.c
Development:SIMNET:MCC:SCC:place.c

place.c implements the user interface for placing vehicle simulators. Table 2.13-20 describes the variables used by place.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
placeCompany	char	Standard C type.
placeTableDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
placeTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
placeColumns	extern array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
placeSelectFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
placeSelectFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
placeSelectDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
placeColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
placeTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
placeTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
placeTableDialog	*DialogNodeDefn*	*Development*:SIMNET:libmac: sequence.h

Table 2.13-20: place.c Variable Information.

2.13.2.2.1 SetUpPlace

SetUpPlace sets up the dialog for placing combat vehicles. The function call is SetUpPlace(). Table 2.13-21 describes the functions called by this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
Called By	
Function	**Where Described**
SetUp	See Section 2.13.8.17.1.

Table 2.13-21: SetUpPlace Information.

2.13.2.2.2 PlaceSelectFetch

PlaceSelectFetch reads the participating organizational units into the Place Select dialog fields to display to the user. The Place Select dialog allows the user to select the company whose simulators are to be placed. The function call is PlaceSelectFetch(dialog). Table 2.13-22 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
DisabieControl	See Section 2.22.1.7.1.	

Table 2.13-22: PlaceSelectFetch Information.

2.13.2.2.3 PlaceSelectEvent

PlaceSelectEvent is called when an event occurs in the Place Select dialog. The Next button is enabled. The function call is PlaceSelectEvent(dialog, itemNo). Table 2.13-23 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button pressed
Calls		
Function	**Where Described**	
EnableControl	See Section 2.22.1.7.2.	

Table 2.13-23: PlaceSelectEvent Information.

2.13.2.2.4 PlaceTableFetch

PlaceTableFetch reads the selected company's simulators into the Place Table dialog fields to display to the user. The function call is PlaceTableFetch(dialog). Table 2.13-24 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
SimTableSetup	See Section 2.22.2.10.1.	

Table 2.13-24: PlaceTableFetch Information.

2.13.2.2.5 PlaceComplete

PlaceComplete is called to take down the Place Table dialog when a selection is successfully placed. The function call is PlaceComplete(sim, success). Table 2.13-25 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sim	pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
success	int	Standard C type.
Calls		
Function	**Where Described**	
UpdateScrollTableEntry	See Section 2.22.1.31.1.	
ZoomDownToTableEntry	See Section 2.13.8.8.3.	

Table 2.13-25: PlaceComplete Information.

2.13.2.2.6 PlaceTableSelect

PlaceTableSelect is called when an entry in the place table is selected. It checks for validity of the selection and pops up the dialog for placing the vehicle. The function call is PlaceTableSelect(defn, row, box, event). Table 2.13-26 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h

Calls	
Function	**Where Described**
ScrollTableRowToEntry	See Section 2.22.1.34.1.
SetScrollTableSelection	See Section 2.22.1.37.2.
SimTableEntry	See Section 2.22.2.10.3.
ShowCaution	See Section 2.22.1.4.1.
ShowVehicleDialog	See Section 2.22.2.9.5.
ZoomUpFromTableEntry	See Section 2.13.8.8.2.

Table 2.13-26: PlaceTableSelect Information.

2.13.2.2.7 PlaceTableEvent

PlaceTableEvent is called when an event occurs in the place table dialog. If the Overview button was pressed, the scroll table is discarded and the overview screen is put up. The item number of the event is returned. The function call is PlaceTableEvent(dialog, itemNo). Table 2.13-27 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button pressed

Calls	
Function	**Where Described**
DisposeScrollTable	See Section 2.22.1.30.2.

Table 2.13-27: PlaceTableEvent Information.

2.13.2.2.8 SimulatorPlaced

SimulatorPlaced notes that a simulator, *sim*, has been placed by another console. If a table of simulators is being displayed, the screen is updated. The function call is SimulatorPlaced(sim). Table 2.13-28 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
Calls		
Function	**Where Described**	
SimTableUpdate	See Section 2.22.2.10.2.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.13.8.6.1.	

Table 2.13-28: SimulatorPlaced Information.

2.13.3 Simulation of Combat Service Support

2.13.3.1 css.c

Development:SIMNET:MCC:SCC:css.c

css.c implements the user interface for specifying the organization of combat service support assets, and the default locations of those assets. Table 2.13-29 describes the variables used by css.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
cssTypeDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
orgTrainsDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
echTrainsDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
unitTrainsDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
ammoTruckTableDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
supplyDepotsDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
umcpDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
supplyRatesDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
depotsConfirmDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cssConfirmDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
supplyDepotsInited	extern char	Standard C type.
cssTypeBuffer	char	Standard C type.
cssTypeFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
cssTypeFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
cssTypeDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
orgTrainsFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
orgTrainsFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
orgTransDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
unitTrainsField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
unitTrainsFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
unitTrainsDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
echTrainsFields	array of CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
echTrainsFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
echTrainsDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cssConfirmFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
cssConfirmDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-29: css.c Variable Information.

2.13.3.1.1 LoadCannedCSS

LoadCannedCSS loads canned CSS initialization data. The function call is LoadCannedCSS(). Table 2.13-30 describes the internal variable used and function called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Calls		
Function	**Where Described**	
StringToMapCoordinates	See Section 2.22.1.26.1.	
Called By		
Function	**Where Described**	
LoadCannedData	See Section 2.13.8.15.2.	

Table 2.13-30: LoadCannedCSS Information.

2.13.3.1.2 CSSTypeFetch

CSSTypeFetch sets up the CSS Types in the dialog fields to display to the user. The radio buttons for initialized CSS types are disabled in the CSS Type dialog. The function call is CSSTypeFetch(state). Table 2.13-31 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
itemType	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
DisableControl	See Section 2.22.1.7.1.	

Table 2.13-31: CSSTypeFetch Information.

2.13.3.1.3 CSSTypeBranch

CSSTypeBranch brings up the correct dialog for the CSS Type selected. A handle to the next dialog to pop up is returned. The function call is CSSTypeBranch(state). Table 2.13-32 describes the parameter used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogSeqState	Development:SIMNET:libmac: sequence.h
Return Values		
Return Value	**Type**	**Meaning**
cssTypeBuffer	pointer to DialogNodeDefn	Handle to next dialog to pop-up.

Table 2.13-32: CSSTypeBranch Information.

2.13.3.1.4 DefaultTrainsEvent

DefaultTrainsEvent is called when a button in one of the trains dialogs is pressed. It determines whether the data entered is valid. It throws the current dialog away and goes to the main screen. It returns the item number of the button pressed. The function call is DefaultTrainsEvent(state, itemNo). Table 2.13-33 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
TruckDefaultAll	See Section 2.13.3.4.2.	

Table 2.13-33: DefaultTrainsEvent Information.

2.13.3.1.5 OrgTrainsBranch

OrgTrainsBranch checks the type of item selected (whether trains are unit or echeloned) and brings up the next dialog depending on the item type. It returns a handle to the next dialog to pop up. The function call is OrgTrainsBranch(state). Table 2.13-34 describes the parameter used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogSeqState	Development:SIMNET:libmac: sequence.h
Return Values		
Return Value	**Type**	**Meaning**
trainsOrganization	pointer to DialogNodeDefn	a handle to the next dialog

Table 2.13-34: OrgTrainsBranch Information.

2.13.3.1.6 UploadCSSParameters

UploadCSSParameters uploads the CSS parameters to the host. This function is only valid when VERSION is APPLETALK. Otherwise it is a dummy function. The function call is UploadCSSParameters(). Table 2.13-35 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cReq	SCCCSSRequest	Development:SIMNET:MCC: SCC:SCC.h
truck	pointer to TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
i	int	Standard C type.
j	int	Standard C type.
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
str	pointer to char	Standard C type.
Calls		
Function	**Where Described**	
ShowWait	See Section 2.22.1.45.1.	
UploadTruckParameters	See Section 2.13.3.6.1.	
ATPPut	See Section 2.22.1.3.3.	
ThrowWait	See Section 2.22.1.45.2.	
Called By		
Function	**Where Described**	
CSSConfirmEvent	See Section 2.13.3.1.8.	

Table 2.13-35: UploadCSSParameters Information.

2.13.3.1.7 CSSConfirmFetch

CSSConfirmFetch reads the choices of the CSS confirmation into the dialog fields to display to the user. The function call is CSSConfirmFetch(state). Table 2.13-36 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
ParamText	Standard Dialog Manager function for Macintosh.	
SetWTitle	Standard Window Manager function for Macintosh.	

Table 2.13-36: CSSConfirmFetch Information.

2.13.3.1.8 CSSConfirmEvent

CSSConfirmEvent is called when a button in the CSS Confirm dialog is pressed. If the OK button was pressed, the CSS parameters are uploaded. The item number of the button pressed is returned, and the overview screen is put up. The function call is CSSConfirmEvent(state, itemNo). Table 2.13-37 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	the OK button was pressed and the CSS parameters are uploaded
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
UploadCSSParameters	See Section 2.13.3.1.6.	
DialogSeqNext	See Section 2.22.1.39.4.	

Table 2.13-37: CSSConfirmEvent Information.

2.13.3.2 depots.c
Development:SIMNET:MCC:SCC:depots.c

depots.c implements the user interface for locating supply depots and establishing ammunition controlled supply rates. Table 2.13-38 describes the variables used by depots.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
supplyDepotsDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
supplyRatesDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
depotsConfirmDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
supplyDepotsInited	char	Standard C type.
supplyDepotsFields	array of CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
supplyDepotsFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
supplyDepotsDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
supplyRatesFields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
supplyRatesFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
supplyRatesDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
depotsConfirmFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
depotsConfirmDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-38: depots.c Variable Information.

2.13.3.2.1 DepotConfirmFetch

DepotConfirmFetch reads the supply depot initialization choices into the Depot Confirm dialog fields to display to the user. The function call is DepotConfirmFetch(state). Table 2.13-39 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
ParamText	Standard Dialog Manager function for Macintosh.	
SetWTitle	Standard Window Manager function for Macintosh.	

Table 2.13-39: DepotConfirmFetch Information.

2.13.3.2.2 DepotConfirmEvent

DepotConfirmEvent is called when a button in the Depot Confirm dialog is pressed. It determines whether the data entered is valid. If the OK button was pressed, the supply depots are initialized, the UMCP and depots are placed into the table of displaceable entities, and the overview screen is put up. Otherwise, the item number of the button pressed is returned. The function call is DepotConfirmEvent(state, itemNo). Table 2.13-40 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCDepotsRequest	Development:SIMNET:MCC:SCC:SCC.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	OK button was pressed and the supply depots are initialized
itemNo	int	resource id of button hit
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	
DialogSeqNext	See Section 2.22.1.39.4.	

Table 2.13-40: DepotsConfirmEvent Information.

2.13.3.3 truckentry.c

Development:SIMNET:MCC:SCC:truckentry.c

truckentry.c implements dialogs for initializing or reconstituting individual supply trucks and maintenance teams. Table 2.13-41 describes the variables used by truckentry.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
currentTruck	pointer to TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
truckBuffer	TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
fuelBuffer	int	Standard C type.
(completionRoutine)()	void function call	Standard
transferWindow	WindowPtr	Development:THINK C:Mac #includes:WindowMgr.h
transferZoomRect	Rect	Development:THINK C:Mac #includes:MacTypes.h
supplyDepot	Manifest	Development:SIMNET.include:resupply.h
truckParty	TransferParty	Development:SIMNET:libsupply:libsupply.h
depotParty	TransferParty	Development:SIMNET:libsupply:libsupply.h
truckAmmoField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
ammoTruckEntryDialog	extern DialogDefn	Development:SIMNET:libmac:dialog.h
fuelTruckEntryDialog	extern DialogDefn	Development:SIMNET:libmac:dialog.h
maintTruckEntryDialog	extern DialogDefn	Development:SIMNET:libmac:dialog.h
truckCancelField	PushButtonFieldDefn	Development:SIMNET:libmac:dialog.h
truckAssignFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
truckSideFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
truckLocationField	CoordinatesFieldDefn	Development:SIMNET:libmac:dialog.h
truckAmmoField	ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
truckFuelField	NumberFieldDefn	Development:SIMNET:libmac:dialog.h
ammoTruckfieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fuelTruckFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
maintTruckFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
ammoTruckEntryDialog	DialogDefn	Development:SIMNET:libmac:dialog.h
fuelTruckEntryDialog	DialogDefn	Development:SIMNET:libmac:dialog.h
maintTruckEntryDialog	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.13-41: truckentry.c Variable Information.

2.13.3.3.1 ShowTruckDialog

ShowTruckDialog shows the dialog for initializing or reconstituting an ammo, fuel, or maintenance truck. The function call is ShowTruckDialog(truck, completion). Table 2.13-42 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
(completion)()	void function call	Standard
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dialog	pointer to array of DialogDefn	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
Called By		
Function	**Where Described**	
TruckTableSelect	See Section 2.13.3.4.11.	
TruckReconstStart	See Section 2.13.5.3.2.	

Table 2.13-42: ShowTruckDialog Information.

2.13.3.3.2 UpdateAmmoLoadTotals

UpdateAmmoLoadTotals updates the weight and volume totals displayed for the current ammunition truck. The function call is UpdateAmmoLoadTotals(). Table 2.13-43 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
weight	long	Standard C type.
volume	long	Standard C type.
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
TotalAmmoCapacity	See Section 2.22.3.9.1.	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
TruckEntryFetch	See Section 2.13.3.3.3.	
TruckEntryEvent	See Section 2.13.3.3.4.	
AmmoTransferComplete	See Section 2.13.3.3.6.	

Table 2.13-43: UpdateAmmoLoadTotals Information.

2.13.3.3.3 TruckEntryFetch

TruckEntryFetch reads the current values of the truck data structures into the dialog fields to display to the user. The vehicle field number is set, radio buttons for companies not participating in this exercise are disabled, the force alignment buttons are labeled and enabled, and the ammo list is displayed for ammo trucks. The function call is TruckEntryFetch(dialog). Table 2.13-44 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
str	array of char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
DisableControl	See Section 2.22.1.7.1.	
LabelForceButtons	See Section 2.22.1.16.1.	
TruckDefaultAlignment	See Section 2.13.3.4.4.	
FillAmmunitionList	See Section 2.22.3.5.1.	
UpdateAmmoLoadTotals	See Section 2.13.3.3.2.	

Table 2.13-44: TruckEntryFetch Information.

2.13.3.3.4 TruckEntryEvent

TruckEntryEvent is called to process data when a button in the truck entry dialog is pressed. The routine determines if the data entered is valid, then updates the dialog. The function call is TruckEntryEvent(dialog, itemNo). Table 2.13-45 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
volume	long	Standard C type.
weight	long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return

Calls	
Function	**Where Described**
CheckMandatoryFields	See Section 2.22.1.13.1
TotalAmmoCapacity	See Section 2.22.3.9.1.
ShowCaution	See Section 2.22.1.4.1.
DisposeScrollTable	See Section 2.22.1.30.2.
TruckDefault	See Section 2.13.3.4.3.
LabelForceButtons	See Section 2.22.1.16.1.
TruckDefaultAlignment	See Section 2.13.3.4.4.
UpdateAmmoList	See Section 2.22.3.5.12.
UpdateAmmoLoadTotals	See Section 2.13.3.3.2.
UpdateDialog	See Section 2.22.1.11.2.

Table 2.13-45: TruckEntryEvent Information.

2.13.3.3.5 AmmoTableSelect

AmmoTableSelect is called when an entry in the Ammo Table is selected. It checks for validity of the selection and brings up the Ammo Transfer dialog. The function call is AmmoTableSelect(defn, row, rect, theEvent). Table 2.13-46 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	*int*	Standard C type.
rect	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h

Calls	
Function	**Where Described**
ShowTransferAmmoDialog	See Section 2.22.3.4.1.
ZoomToWindow	See Section 2.22.1.47.4.
ShowWindow	Standard Window Manager function for Macintosh.

Table 2.13-46: AmmoTableSelect Information.

2.13.3.3.6 AmmoTransferComplete

AmmoTransferComplete is called to take down the Ammo Transfer dialog when the transfer has been completed. The ammo list and ammo load totals are updated. The function call is AmmoTransferComplete(). Table 2.13-47 describes the functions called using this function.

Calls	
Function	**Where Described**
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
UpdateAmmoList	See Section 2.22.3.5.12.
UpdateAmmoLoadTotals	See Section 2.13.3.3.2.

Table 2.13-47: AmmoTransferComplete Information.

2.13.3.4 trucktable.c

Development:SIMNET:MCC:SCC:trucktable.c

trucktable.c implements dialogs displaying tables of supply trucks. Table 2.13-48 describes the variables used by trucktable.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
truckDistribution	array of LongPt	Development:SIMNET:libmac:longpt.h
defaultAmmoLoad[numberBn977Vehicles][numberAmmoVarieties]	matrix of short	Standard C type.
fuelTruckTableDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
maintTruckTableDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
supplyDepotsDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
ammoTruckTableField	extern FixTableFieldDefn	Development:SIMNET:libmac:table.h
fuelTruckTableField	extern FixTableFieldDefn	Development:SIMNET:libmac:table.h
maintTruckField	extern FixTableFieldDefn	Development:SIMNET:libmac:table.h
ammoTruckTableColumns	extern array of FixTableColumnDefn	Development:SIMNET:libmac:table.h
fuelTruckTableColumns	extern array of FixTableColumnDefn	Development:SIMNET:libmac:table.h
maintTruckTableColumns	extern array of FixTableColumnDefn	Development:SIMNET:libmac:table.h
truckTable	pointer to FixTableFieldDefn	Development:SIMNET:libmac:table.h
selectRow	short	Standard C type.
byAmmoType	char	Standard C type.

(Table 2.13-48 is continued on the following page.)

Variable	Type	Where Typedef Declared
ammoTruckTableField	FixTableFieldDefn	Development:SIMNET:libmac:table.h
ammoTruckLoadFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
ammoTruckTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
ammoTruckTableDialog	DialogNodeDefn	Development:SIMNET:libmac:sequence.h
fuelTruckTableColumns	array of FixTableColumnDefn	Development:SIMNET:libmac:table.h
fuelTruckTableField	FixTableFieldDefn	Development:SIMNET:libmac:table.h
fuelTruckTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fuelTruckTableDialog	DialogNodeDefn	Development:SIMNET:libmac:sequence.h
maintTruckTableColumns	array of FixTableColumnDefn	Development:SIMNET:libmac:table.h
maintTruckField	FixTableFieldDefn	Development:SIMNET:libmac:table.h
maintTruckTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
cssConfirmDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
maintTruckTableDialog	DialogNodeDefn	Development:SIMNET:libmac:sequence.h

Table 2.13-48: trucktable.c Variable Information.

2.13.3.4.1 SetUpTrucks

SetUpTrucks initializes the truck tables. The function call is SetUpTrucks(). Table 2.13-49 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
Called By	
Function	**Where Described**
SetUp	See Section 2.13.8.17.1.

Table 2.13-49: SetUpTrucks Information.

2.13.3.4.2 TruckDefaultAll

TruckDefaultAll gives default values to all truck parameters for the truck pointed to by *truck*. The function call is TruckDefaultAll(). Table 2.13-50 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Calls		
Function	**Where Described**	
TruckDefault	See Section 2.13.3.4.3.	
Called By		
Function	**Where Described**	
DefaultTrainsEvent	See Section 2.13.3.1.4.	

Table 2.13-50: TruckDefaultAll Information.

2.13.3.4.3 TruckDefault

TruckDefault restores the default values to the specified truck's initial parameters. The function call is TruckDefault(truck). Table 2.13-51 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
pt	LongPt	Development:SIMNET:libmac:longpt.h
Calls		
Function	**Where Described**	
TruckDefaultAlignment	See Section 2.13.3.4.4.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
Called By		
Function	**Where Described**	
TruckDefaultAll	See Section 2.13.3.4.2.	
TruckEntryEvent	See Section 2.13.3.3.4.	

Table 2.13-51: TruckDefault Information.

2.13.3.4.4 TruckDefaultAlignment

TruckDefaultAlignment returns the default alignment of *truck* according to the company. The function call is TruckDefaultAlignment(truck). Table 2.13-52 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to Truck Descriptor	Defvelopment:SIMNET:MCC:SCC:truck.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
forceID	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
forceID	int	Force to which *truck* belongs.
Called By		
Function	**Where Described**	
TruckDefault	See Section 2.13.3.4.3.	
TruckEntryFetch	See Section 2.13.3.3.3.	
TruckEntryEvent	See Section 2.13.3 3.4.	

Table 2.13-52: TruckDefaultAlignment Information.

2.13.3.4.5 AmmoTruckTableFetch

AmmoTruckTableFetch reads the current values of the Ammo Truck Table into the dialog fields to display to the user. The Ammo Truck Table displays the trucks' initial parameters. The function call is AmmoTruckTableFetch(dialog). Table 2.13-53 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	

Table 2.13-53: AmmoTruckTableFetch Information.

2.13.3.4.6 AmmoTruckTableEvent

AmmoTruckTableEvent is called when a button in the ammo truck table dialog is pressed. The validity of the choice is checked, and the data entered is processed. It throws the current dialog away and goes to the main screen. The item number of the button pressed is returned. The function call is AmmoTruckTableEvent(state, itemNo). Table 2.13-54 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to Dialog State	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
rect	Rect	Development:THINK C: Mac #includes:MacTypes.h
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
GetDItem	Standard Dialog Manager function for Macintosh.	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	

Table 2.13-54: AmmoTruckTableEvent Information.

2.13.3.4.7 FuelTruckTableFetch

FuelTruckTableFetch sets the text of the fuel truck table into the dialog fields to display to the user.The function call is FuelTruckTableFetch(dialog). Table 2.13-55 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to Dialog State	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	

Table 2.13-55: FuelTruckTableFetch Information.

2.13.3.4.8 MaintTruckTableFetch

MaintTruckTableFetch sets the text of the maintenance truck table into the dialog fields to display to the user.The function call is MaintTruckTableFetch(dialog). Table 2.13-56 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to Dialog State	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	

Table 2.13-56: MaintTruckTableFetch Information.

2.13.3.4.9 TruckTableComplete

TruckTableComplete is called when an entry in the Truck Table has been successfully completed. The entry dialog is taken down, the Truck Table is updated, and the Truck Table dialog is put up. The function call is TruckTableComplete(success). Table 2.13-57 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
success	int	Standard C type.
Calls		
Function	**Where Described**	
UpdateFixTableRow	See Section 2.22.1.42.2.	
ZoomDownToTableEntry	See Section 2.13.8.8.3.	
ThrowDialog	See Section 2.22.1.11.3.	

Table 2.13-57: TruckTableComplete Information.

2.13.3.4.10 TruckTableSelect

TruckTableSelect is called when an entry in a truck table is selected. The selected entry is highlighted, and a dialog is opened for it. The function call is TruckTableSelect(defn, row, box). Table 2.13-58 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
buf	register pointer to TruckTableData	Development:SIMNET:MCC: SCC:truck.h
r	Rect	Development:THINK C: Mac #includes:MacTypes.h
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h

Calls	
Function	**Where Described**
GetPort	Standard Quickdraw function for Macintosh.
SetPort	Standard Quickdraw function for Macintosh.
FixTableRowRect	See Section 2.22.1.42.5.
InvertRect	Standard Quickdraw function for Macintosh.
ShowTruckDialog	See Section 2.13.3.3.1.
ZoomUpFromTableEntry	See Section 2.13.8.8.2.

Table 2.13-58: TruckTableSelect Information.

2.13.3.4.11 TruckTableHilite

TruckTableHilite changes the appearance of the given selection so the user knows the item is selected. The text is inverted. The function call is TruckTableHilite(defn, row, box). Table 2.13-59 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Calls	
Function	**Where Described**
InvertRect	Standard Quickdraw function for Macintosh.

Table 2.13-59: TruckTableHilite Information.

2.13.3.4.12 TruckDrawNumber

TruckDrawNumber fills in the truck number field in the specified column in the table to display to the user. The function call is TruckDrawNumber(defn, col, row). Table 2.13-60 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal C onversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
CharWidth	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	

Table 2.13-60: TruckDrawNumber Information.

2.13.3.4.13 TruckDrawAssignment

TruckDrawAssignment fills in the company assignment field of the specified column in the table to display to the user. The function call is TruckDrawAssignment(defn, col, row). Table 2.13-61 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
pt	Point	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
GetPen	Standard Quickdraw function for Macintosh.	
MoveTo	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	
DrawChar	Standard Quickdraw function for Macintosh.	

Table 2.13-61: TruckDrawAssignment Information.

2.13.3.4.14 TruckDrawLocation

TruckDrawLocation fills in the truck location field for the specified column in the table to display to the user. The function call is TruckDrawLocation(defn, col, row). Table 2.13-62 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.13-62: TruckDrawLocation Information.

2.13.3.4.15 TruckDrawFuelLoad

TruckDrawFuelLoad fills in the truck fuel load for the specified column in the table to display to the user. The function call is TruckDrawFuelLoad(defn, col, row). Table 2.13-63 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
str	array of char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-63: TruckDrawFuelLoad Information.

2.13.3.4.16 TruckDrawAmmoLoad

TruckDrawAmmoLoad fills in the truck ammo load field for the specified column in the table to display to the user. The function call is TruckDrawAmmoLoad(table, column, row). Table 2.13-64 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
table	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
column	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	pointer to TruckDescriptor	Development SIMNET:MCC: SCC:truck.h
rect	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
FixTableRowRect	See Section 2.22.1.42.5.	
DisplayAmmoLoadSummary	See Section 2.22.3.7.1.	

Table 2.13-64: TruckDrawAmmoLoad Information.

2.13.3.5 truck.h

Development:SIMNET:MCC:SCC:truck.h

truck.h is the header file associated with the following file, truck.c , in Section 2.13.3.6. Table 2.13-65 describes the variables used by truck.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
ammoTrucks	extern array of TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
fuelTrucks	extern array of TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
maintTrucks	extern array of TruckDescriptor	Development:SIMNET:MCC: SCC:truck.h
ammoTruckData	extern TruckTableData	Development:SIMNET:MCC: SCC:truck.h
fuelTruckData	extern TruckTableData	Development:SIMNET:MCC: SCC:truck.h
maintTruckData	extern TruckTableData	Development:SIMNET:MCC: SCC:truck.h

Table 2.13-65: truck.h Variable Informaiton.

2.13.3.6 truck.c
Development:SIMNET:MCC:SCC:truck.c

truck.c contains various functions related to the SCC application's support of Combat vehicles. Table 2.13-66 describes the variables used by truck.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
ammoTruckEntryDialog	extern DialogDefn	Development:SIMNET:libmac:dialog.h
fuelTruckEntryDialog	extern DialogDefn	Development:SIMNET:libmac:dialog.h
maintTruckEntryDialog	extern DialogDefn	Development:SIMNET:libmac:dialog.h
ammoTrucks	array of numberBnM977Vehicles TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
ammoTruckData	TruckTableData	Development:SIMNET:MCC:SCC:truck.h
fuelTrucks	array of numberBnM97Vehicles TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
fuelTruckData	TruckTableData	Development:SIMNET:MCC:SCC:truck.h
maintTrucks	array of numberBnRepairVehicles TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
maintTruckData	TruckTableData	Development:SIMNET:MCC:SCC:truck.h

Table 2.13-66: truck.c Variable Informaiton.

2.13.3.6.1 UploadTruckParameters

UploadTruckParameters tells the host a truck's location and load. *truck* specifies the truck. The function call is UploadTruckParameters(truck). Table 2.13-67 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCTruckInitRequest	Development:SIMNET:MCC:SCC:SCC.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	

Called By	
Function	**Where Described**
TruckReconstComplete	See Section 2.13.5.3.3.
UploadCSSParameters	See Section 2.13.3.1.6.

Table 2.13-67: UploadTruckParameters Information.

2.13.3.6.2 DownloadTruckParameters

DownloadTruckParameters obtains from the host information about a truck's current location and load. The function call is DownloadTruckParameters(truck). Table 2.13-68 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCTruckQueryRequest	Development:SIMNET:MCC:SCC:SCC.h
rsp	SCCTruckQueryResponse	Development:SIMNET:MCC:SCC:SCC.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
Called By		
Function	**Where Described**	
TruckReconstStart	See Section 2.13.5.3.2.	

Table 2.13-68: DownloadTruckParameters Information.

2.13.4 Simulation of Close Air and Fire Support

2.13.4.1 cas.c

Development:SIMNET:MCC:SCC:cas.c

cas.c implements the user interface for initializing the close air support simulation and for allotting additional sorties for close air support. Table 2.13-69 describes the variables used by cas.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
daySortiesAllocated	DateTimeRec	Development:THINK C: Mac #includes:OSUtil.h
totalSorties	int	Standard C type.
preplannedSorties	int	Standard C type.
addlTotalSorties	int	Standard C type.
addPreplannedSorties	int	Standard C type.
initCASFields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
initCASFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
initCASDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
moreSortiesFields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
moreSortiesFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
moreSortiesDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-69: cas.c Variable Information.

2.13.4.1.1 LoadCannedCAS

LoadCannedCAS loads canned CAS initialization data. The function call is LoadCannedCAS(). Table 2.13-70 describes this function.

Called By	
Function	**Where Described**
LoadCannedData	See Section 2.13.8.15.2.

Table 2.13-70: LoadCannedCAS Information.

2.13.4.1.2 UploadCASParameters

UploadCASParameters upload the Close Air Support parameters to the host. This function is only compiled if VERSION is APPLETALK. Otherwise, it is a dummy function.The function call is UploadCASParameters(). Table 2.13-71 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCCASRequest	Development:SIMNET:MCC: SCC:SCC.h
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	

Called By	
Function	**Where Described**
InitCASEvent	See Section 2.13.4.1.3.
MoreSortiesEvent	See Section 2.13.4.1.5.

Table 2.13-71: UploadCASParameters Information.

2.13.4.1.3 InitCASEvent

InitCASEvent is called when a button in the initialize CAS dialog is pressed. The Initialize CAS dialog permits the establishment of sortie limits. If the OK button was pressed, the CAS parameters are uploaded . The item number of the button pressed is returned. The function call is InitCASEvent(state, itemNo). Table 2.13-72 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
ShowCaution	See Section 2.22.1.4.1.	
UploadCASParameters	See Section 2.13.4.1.2.	
GetTime	Standard Operating System Utility function for Macintosh.	

Table 2.13-72: InitCASEvent Information.

2.13.4.1.4 MoreSortiesFetch

MoreSortiesFetch reads the current values of the sorties limits data structures into the More Sorties dialog fields to display to the user. The More Sorties dialog permits more sorties to be allocated to CAS. The function call is MoreSortiesFetch(state). Table 2.13-73 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
now	DateTimeRec	Development:THINK C: Mac #includes:OSUtil.h
str	array of 20 char	Standard C type.

Calls	
Function	**Where Described**
GetTime	Standard Operating System Utility function for Macintosh.
SetText	See Section 2.22.1.11.8.

Table 2.13-73: MoreSortiesFetch Information.

2.13.4.1.5 MoreSortiesEvent

MoreSortiesEvent is called when an event occurs in the More Sorties dialog. If the OK button is pressed, the CAS parameters are uploaded, and the total number of sorties is calculated. The item number of the button pressed is returned. The function call is MoreSortiesEvent(state, itemNo). Table 2.13-74 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
ShowCaution	See Section 2.22.1.4.1.	
UploadCASParameters	See Section 2.13.4.1.2.	

Table 2.13-74: MoreSortiesEvent Information.

2.13.4.2 fse.c

Development:SIMNET:MCC:SCC:fse.c

fse.c implements the user interface for initializing and resupplying fire support elements simulated by the MCC system. Table 2.13-75 describes the variables used by fse.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
initMortarDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
fseConfirmDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
numberHowitzerBatteries	extern int	Standard C type.
nthBattery	short	Standard C type.
batteryOrdinals	array of Str255	Development:THINK C: Mac #includes:MacTypes.h
gunData	array of maxNumberBatteries GunData	Development:SIMNET:MCC: SCC:fse.c
currentBattery	short	Standard C type.
initArty1LocField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
initArty1Fields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
initArty1FieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
initArty1Dialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
initArty2LocField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
initArty2Fields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
initArty2FieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
initArty2Dialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
initArty3LocField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
initArty3Fields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
initArty3FieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
initArty3Dialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
initMortLocField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
initMortFields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
initMortFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
initMortDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
fseConfirmFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
fseConfirmDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
reconstBtryLocField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h

(Table 2.13-75 is continued on the following page.)

Variable	Type	Where Typedef Declared
reconstBtryFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
reconstBtryDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-75: fse.c Variable Information.

2.13.4.2.1 LoadCannedFSE

LoadCannedFSE loads canned FSE data. The function call is LoadCannedFSE(). Table 2.13-76 describes the functions called using this function.

Calls	
Function	**Where Described**
StringToMapCoordinates	See Section 2.22.1.26.1.
Called By	
Function	**Where Described**
LoadCannedData	See Section 2.13.8.15.2.

Table 2.13-76: LoadCannedFSE Information.

2.13.4.2.2 InitArtyFetch

InitArtyFetch reads the current values of the artillery data structures into the dialog fields to display to the user. The Initialize Artillery dialog accepts positions and supplies for batteries. There may be up to three Init Arty dialogs. The function call is InitArtyFetch(state). Table 2.13-77 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
ParamText	Standard Dialog Manager function for Macintosh.	

Table 2.13-77: InitArtyFetch Information.

2.13.4.2.3 InitArtyEvent

InitArtyEvent is called when an event occurs in the Init Arty dialog. It determines whether the data entered is valid. It throws the current dialog away and goes to the main screen. It returns the item number of the button pressed. The function call is InitArtyEvent(dialog, itemNo). Table 2.13-78 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	so nothing upon return
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	

Table 2.13-78: InitArtyEvent Information.

2.13.4.2.4 InitArtyBranch

InitArtyBranch determines the next dialog to pop up when an item from a list is selected. It checks the type of item selected and brings up the next dialog, depending on the item type. It returns a handle to the next dialog to pop up. The function call is InitArtyBranch(state). Table 2.13-79 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogSeqState	Development:SIMNET:libmac: sequence.h
Return Values		
Return Value	**Type**	**Meaning**
&initMortDialog	pointer to DialogNodeDefn	handle of the Mortors Dialog
&initArty2Dialog	pointer to DialogNodeDefn	handle of the second Artillery Dialog
&initArty3Dialog	pointer to DialogNodeDefn	handle of the third Artillery Dialog

Table 2.13-79: InitArtyBranch Information.

2.13.4.2.5 UploadFSEParameters

UploadFSEParameters uploads the FSE initialization parameters. This function will only compile if VERSION is APPLETALK. Otherwise, it is a dummy function. The function call is UploadFSEParameters(). Table 2.13-80 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCFSERequest	Development:SIMNET:MCC: SCC:SCC.h
i	short	Standard C type.
j	short	Standard C type.

Calls	
Function	**Where Described**
SetCursor	Standard Quickdraw function for Macintosh.
ATPPut	See Section 2.22.1.3.3.
Called By	
Function	**Where Described**
FSEConfirmEvent	See Section 2.13.4.2.7.

Table 2.13-80: UploadFSEParameters Information.

2.13.4.2.6 FSEConfirmFetch

FSEConfirmFetch reads the FSE confirmation choices into the FSE Confirm dialog fields to display to the user. The function call is FSEConfirmFetch(dialog). Table 2.13-81 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
ParamText	Standard Dialog Manager function for Macintosh.	
SetWTitle	Standard Window Manager function for Macintosh.	

Table 2.13-81: FSEConfirmFetch Information.

2.13.4.2.7 FSEConfirmEvent

FSEConfirmEvent is called when an event occurs in the FSE Confirm dialog. The validity of the data is checked. If the OK button was pressed, the FSE parameters are uploaded, and the guns are added to the table of displaceable entities. It throws the current dialog away and goes to the next dialog in the sequence. It returns the item number of the button pressed. The function call is FSEConfirmEvent(dialog, itemNo). Table 2.13-82 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
dtIndex	int	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of button hit
Calls		
Function	**Where Described**	
UploadFSEParameters	See Section 2.13.4.2.5.	
DialogSeqNext	See Section 2.22.1.39.4.	

Table 2.13-82: FSEConfirmEvent Information.

2.13.4.2.8 UploadBtryParameters

UploadBtryParameters uploads new battery parameters. This function will only compile if VERSION is APPLETALK. Otherwise, it is a dummy function. The function call is UploadBtryParameters(battery). Table 2.13-83 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	register int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCArtyReconstRequest	Development:SIMNET:MCC:SCC:SCC.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
BtryReconstEvent	See Section 2.13.4.2.11.	

Table 2.13-83: UploadBtryParameters Information.

2.13.4.2.9 DownloadBtryParameters

DownloadBtryParameters downloads battery parameters from host. This function will only compile if VERSION is APPLETALK. Otherwise, it is a dummy function. The function call is DownloadBtryParameters(battery). Table 2.13-84 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	register int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCArtyQueryRequest	Development:SIMNET:MCC:SCC:SCC.h
rsp	SCCArtyQueryResponse	Development:SIMNET:MCC:SCC:SCC.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
Called By		
Function	**Where Described**	
ShowBtryReconstDialog	See Section 2.13.4.2.10.	

Table 2.13-84: DownloadBtryParameters Information.

2.13.4.2.10 ShowBtryReconstDialog

ShowBtryReconstDialog determines which battery is being reconstituted, downloads information about the battery, and puts up the dialog allowing the reconstitution of mortars. The function call is ShowBtryReconstDialog(battery). Table 2.13-85 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	register int	Standard C type.
Calls		
Function	**Where Described**	
DownloadBtryParameters	See Section 2.13.4.2.9.	
DialogSeqNext	See Section 2.22.1.39.4.	
Called By		
Function	**Where Described**	
ReconstEvent	See Section 2.13.5.3.7.	

Table 2.13-85: ShowBtryReconstDialog Information.

2.13.4.2.11 BtryReconstEvent

BtryReconstEvent is called when a button in the Battery Reconstitute dialog is pressed. It determines whether the data entered is valid. If the OK button was pressed, the battery parameters are uploaded. If the Cancel button was pressed, the current dialog is thrown away and the overview screen is put up. The item number of the button pressed is returned. The function call is BtryReconstEvent(dialog, itemNo). Table 2.13-86 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
dtIndex	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
okBItem	int	resource id of OK button
itemNo	int	resource id of button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
UploadBtryParameters	See Section 2.13.4.2.8.	

Table 2.13-86: BtryReconstEvent Information.

2.13.4.2.12 GetGunAzimuth

GetGunAzimuth returns the gun azimuth of a battery. The function call is GetGunAzimuth(battery). Table 2.13-87 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
gunData[battery].azimuth	int	Gun azimuth of a battery.
Called By		
Function	**Where Described**	
NetworkArrived	See Section 2.13.8.10.22.	

Table 2.13-87: GetGunAzimuth Information.

2.13.4.3 cew.h
Development:SIMNET:MCC:SCC:cew.h

cew.h is the header file associated with cew.c, in Section 2.13.4.4. Table 2.13-88 describes the variables used by cew.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
cewAssetNumber	extern short	Standard C type.
cewAssets	extern array of maxCEWAssets CEWVehicle	Development:SIMNET:MCC:SCC:cew.h
cewReconstAsset	extern short	Standard C type.
cewPlatoonVehicleName	extern array of 20 char	Standard C type.
cewGEMSSName	extern array of 20 char	Standard C type.
cewM57Name	extern array of 20 char	Standard C type.
cewLineChargeName	extern array of 20 char	Standard C type.

Table 2.13-88: cew.h Variable Information.

2.13.4.4 cew.c
Development:SIMNET:MCC:SCC:cew.c

cew.c implements the Combat Engineering resource allocation. Table 2.13-89 describes the variables used by cew.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
initOverviewDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cewAllocDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cewPlaceDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cewConfirmDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cewDefaultLocation	MapCoordinates	Development:SIMNET:libmac: map.h
cewPlatoonNumber	short	Standard C type.
cewGEMSSNumber	short	Standard C type.
cewLineCharger	short	Standard C type.
cewM57Number	short	Standard C type.
cewAssetNumber	short	Standard C type.
cewReconstAsset	short	Standard C type.
cewPlatoonVehicleName	array of 20 char	Standard C type.
cewGEMSSName	array of 20 char	Standard C type.
cewM57Name	array of 20 char	Standard C type.
cewLineChargerName	array of 20 char	Standard C type.
cewAssets	array of maxCEWAssets CEWVehicle	Development:SIMNET:MCC: SCC:cew.h
cewAllocFields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
cewDefaultLocationField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
cewAllocFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
cewAllocDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cewConfirmFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
cewConfirmDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cewReconstLocation	MapCoordinates	Development:SIMNET:libmac: map.h
cewReconstLocationField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
cewReconstFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
cewReconstDailog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-89: cew.c Variable Information.

2.13.4.4.1 LoadDefaultCEWParameters

LoadDefaultCEWParameters loads in default CEW parameters. The function call is LoadDefaultCEWParameters(). Table 2.13-90 describes the functions called using this function.

Calls	
Function	**Where Described**
StringToMapCoordinates	See Section 2.22.1.26.1.
Called By	
Function	**Where Described**
CEWAllocFetch	See Section 2.13.4.4.2.

Table 2.13-90: LoadDefaultCEWParameters Information.

2.13.4.4.2 CEWAllocFetch

CEWAllocFetch reads the current values of the CEW resource allocations into the dialog fields to display to the user. The function call is CEWAllocFetch(state). Table 2.13-100 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
LoadDefaultCEWParameters	See Section 2.13.4.4.1.	
SellText	Standard Dialog Manager function for Macintosh.	

Table 2.13-100: CEWAllocFetch Information.

2.13.4.4.3 CEWAllocEvent

CEWAllocEvent is called when a button in the CEW Allocation dialog is pressed. If the Next button was pressed, the numbers are set, the assets are set up, and the gemss, M57 and live charges are initialized. The function call is CEWAllocEvent(dialog, itemNo). Table 2.13-101 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	short	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
counter	short	Standard C type.
asset	short	Standard C type.
p1	short	Standard C type.
p2	short	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	short	do nothing upon return
itemNo	short	resource id of button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	

Table 2.13-101: CEWAllocEvent Information.

2.13.4.4.4 UploadCEWParameters

UploadCEWParameters uploads CEW parameter information to the host. The function call is UploadCEWParameters(). Table 2.13-102 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	double	Standard C type.
asset	short	Standard C type.
ereq	SCCCEWStartRequest	Development:SIMNET:MCC:SCC:SCC.h
req	SCCCEWInitRequest	Development:SIMNET:MCC:SCC:SCC.h
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
CEWConfirmFetch	See Section 2.13.4.4.6.	

Table 2.13-102: UploadCEWParameters Information.

2.13.4.4.5 CEWConfirmFetch

CEWConfirmFetch reads the CEW confirmation choices into the dialog fields to display to the user. The function call is CEWConfirmFetch(dialog). Table 2.13-103 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
ParamText	Standard Dialog Manager function for Macintosh.	
SetWTitle	Standard Window Manager function for Macintosh.	

Table 2.13-103: CEWConfirmFetch Information.

2.13.4.4.6 CEWConfirmEvent

CEWConfirmEvent is called when a button in the CEW confirm dialog is pressed. If the OK button was pressed, the CEW paramters are uploaded to the host , and the Combat Engineering option is turned off in the overview screen. The function call is CEWConfirmEvent(diualog, itemNo). Table 2.13-104 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ds	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Return Values		
Return Value	**Type**	**Meaning**
cancelBItem	short	resource id of the Cancel button
itemNo	short	resource id of the OK button
Calls		
Function	**Where Described**	
ShowWait	See Section 2.22.1.45.1.	
UploadCEWParameters	See Section 2.13.4.4.4.	
ThrowWait	See Section 2.22.1.45.2.	

Table 2.13-104: CEWConfirmEvent Information.

2.13.4.4.7 CEWReconstFetch

CEWReconstFetch is called in order to fill in the name of the vehicle selected and its current location before the CEW Reconstitute dialog is put up on the screen. The function call is CEWReconstFetch(state). Table 2.13-105 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
buffer	array of 100 char	Standard C type.
lpt	LongPt	Development:SIMNET:libmac: longpt.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
PointToMapCoordinates	See Section 2.22.1.26.2.	
ParamText	Standard Dialog Manager function for Macintosh.	

Table 2.13-105: CEWReconstFetch Information.

2.13.4.4.8 CEWReconstEvent

CEWReconstEvent is called when an event occurs in the CEW Reconstitute dialog. The validity of the data is checked. If the Next button was pressed, it gets the location and tells the host to reconstitute the asset. The item number of the button pressed is returned. The function call is CEWReconstEvent(dialog, itemNo). Table 2.13-106 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCCEWReconstRequest	Development:SIMNET:MCC: SCC:SCC.h
Return Values		
Return Value	**Type**	**Meaning**
0	short	do nothing upon return
nextBItem	short	CEW cancel button was hit
itemNo	short	resource id of the button hit

Calls	
Function	**Where Described**
CheckMandatoryFields	See Section 2.22.1.13.1.
ATPPut	See Section 2.22.1.3.3.

Table 2.13-106: CEWReconstEvent Information.

2.13.4.4.9 UpdateCEWAssetLocation

UpdateCEWAssetLocation gets the latest location of *asset* (the CEW asset) and sets the location field in the dialog. The function call is UpdateCEWAssetLocation(asset). Table 2.13-107 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
asset	short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCCEWQueryRequest	Development:SIMNET:MCC:SCC:SCC.h
rsp	SCCCEWQueryResponse	Development:SIMNET:MCC:SCC:SCC.h
counter	short	Standard C type.
lpt	LongPt	Development:SIMNET:libmac:longpt.h
i	short	Standard C type.
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
ShowCEWReconstDialog	See Section 2.13.4.4.10.	

Table 2.13-107: UpdateCEWAssetLocation Information.

2.13.4.4.10 ShowCEWReconstDialog

ShowCEWReconstDialog puts up the CEW Reconstitute dialog. The function call is ShowCEWReconstDialog(asset). Table 2.13-108 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
asset	short	Standard C type.

Calls	
Function	**Where Described**
UpdateCEWAssetLocation	See Section 2.13.4.4.9.
DialogSeqNext	See Section 2.22.1.39.4.
Called By	
Function	**Where Described**
ReconstEvent	See Section 2.13.5.3.7.

Table 2.13-108: ShowCEWReconstDialog Information.

2.13.5 Reconstitution of Simulated Vehicles

2.13.5.1 reconscr.h

Development:SIMNET:MCC:SCC:reconscr.h

reconscr.h is the header file associated with the following two files, reconscr.c and reconsct.c. This file contains definitions needed to implement scrolling tables in the reconstitution dialog. Table 2.13 109 describes the variables used by reconscr.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
reconstEntityNames	extern pointer to array of char	Standard C type.

Table 2.13-109: reconscr.h Variable Information.

2.13.5.2 reconscr.c

Development:SIMNET:MCC:SCC:reconscr.c

reconscr.c implements a scrolling table of elements that may be reconstituted by the BattleMaster. Table 2.13-110 describes the variables used by reconscr.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
reconstEntityNames	pointer to array of char	Standard C type.

Table 2.13-110: reconscr.c Variable Information.

2.13.5.2.1 NewReconstElement

NewReconstElement allocates memory for a new element in the reconstitution table, and sets up fields. The function call is NewReconstElement(key). Table 2.13-111 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
key	short	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
element	register ReconstElementHandle	Development:SIMNET:MCC: SCC:reconscr.h
Return Values		
Return Value	**Type**	**Meaning**
element	ReconstElementHandle	Handle to the data structure describing the entry that was just created.
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
Called By		
Function	**Where Described**	
ReconstFetch	See Section 2.13.5.3.4.	

Table 2.13-111: NewReconstElement Information.

2.13.5.2.2 NewReconstVehicle

NewReconstVehicle allocates memory for a new vehicle entry in the reconstitution table, and sets up fields. The function call is NewReconstVehicle(type, vehicleNumber). Table 2.13-112 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	short	Standard C type.
vehicleNumber	short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
vehicle	register ReconstVehicleHandle	Development:SIMNET:MCC: SCC:reconscr.h
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
Return Values		
Return Value	**Type**	**Meaning**
vehicle	ReconstVehicleHandle	Handle to the data structure describing the entry that was just created.
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	

Called By	
Function	**Where Described**
ReconstElementSelect	See Section 2.13.5.3.5.

Table 2.13-112: NewReconstVehicle Information.

2.13.5.2.3 DrawReconstElement

DrawReconstElement fills in element fields of the specified column in the reconstitution table to display to the user. The function call is DrawReconstElement(defn, col, entry, box). Table 2.13-113 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
Move	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-113: DrawReconstElement Information.

2.13.5.2.4 DrawReconstVehicle

DrawReconstVehicle fills in vehicle fields of the specified column in the reconstitution table to display to the user. The function call is DrawReconstVehicle(defn, col, entry, box). Table 2.13-114 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	register pointer to SimDescriptor	Development:SIMNET:MCC:libsim:libsim.h
vehicle	register ReconstVehicleHandle	Development:SIMNET:MCC:SCC:reconscr.h
cStr	array of 30 char	Standard C type.
pStr	array of 30 char	Standard C type.
Calls		
Function	**Where Described**	
Move	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-114: DrawReconstVehicle Information.

2.13.5.3 reconst.c

Development:SIMNET:MCC:SCC:reconst.c

reconst.c implements a scrolling table of elements that may be reconstituted by the BattleMaster. Table 2.13-115 describes the variables used by reconst.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
reconstTOCDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
reconstALOCDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
reconstVehicleColumn	extern ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
numberHowitzerBatteries	extern int	Standard C type.
selectedVehicle	short	Standard C type.
selectedVehicleType	short	Standard C type.
reconstSelection	short	Standard C type.
reconstElementTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
reconstVehicleTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
vehicleTableHeading	array of 30 char	Standard C type.
reconstOverviewField	PushButtonFieldDefn	Development:SIMNET:libmac:dialog.h
reconstElementColumn	ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
reconstVehicleColumn	ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
reconstElementTableField	ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
reconstVehicleTableField	ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
reconstFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
reconstDialog	DialogNodeDefn	Development:SIMNET:libmac:sequence.h

Table 2.13-115: reconst.c Variable Information.

2.13.5.3.1 SimReconstComplete

SimReconstComplete is called to bring up the BatterMaster's overview dialog. The function call is SimReconstComplete(sim, success). Table 2.13-116 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sim	pointer to SimDescriptor	Development:SIMNET:MCC:libsim:libsim.h
success	int	Standard C type.
Calls		
Function	**Where Described**	
DisposeScrollTable	See Section 2.22.1.30.2.	
DialogSeqNext	See Section 2.22.1.39.4.	

Table 2.13-116: SimReconstComplete Information.

2.13.5.3.2 TruckReconstStart

TruckReconstStart downloads the latest status of *truck* and puts up the truck dialog in order to modify the status. The function call is TruckReconstStart(truck). Table 2.13-117 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
DownloadTruckParameters	See Section 2.13.3.6.2.	
ShowTruckDialog	See Section 2.13.3.3.1.	
Called By		
Function	**Where Described**	
ReconstEvent	See Section 2.13.5.3.7.	

Table 2.13-117: TruckReconstStart Information.

2.13.5.3.3 TruckReconstComplete

TruckReconstComplete is called to check the truck reconstitution for successful completion. The reconst dialog is taken down and the BattleMaster's overview dialog is put up. The function call is TruckReconstComplete(truck, success). Table 2.13-118 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckDescriptor	Development:SIMNET:MCC:SCC:truck.h
success	int	Standard C type.
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
UploadTruckParameters	See Section 2.13.3.6.1.	
ThrowDialog	See Section 2.22.1.11.3.	
DisposeScrollTable	See Section 2.22.1.30.2.	
DialogSeqNext	See Section 2.22.1.39.4.	

Table 2.13-118: TruckReconstComplete Information.

2.13.5.3.4 ReconstFetch

ReconstFetch builds the recontst element table into dialog fields to display to the user. The function call is ReconstFetch(dialog). Table 2.13-119 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
element	register ReconstElementHandle	Development:SIMNET:MCC:SCC:reconscr.h
vehicle	register ReconstVehicleHandle	Development:SIMNET:MCC:SCC:reconscr.h
sim	register pointer to SimDescriptor	Development:SIMNET:MCC:libsim:libsim.h
i	register int	Standard C type.
first	int	Standard C type.
t1	ScrollTableEntryHandle	Development:SIMNET:libmac:scroll.h
t2	ScrollTableEntryHandle	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
NewReconstElement	See Section 2.13.5.2.1.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
ReconstElementSelect	See Section 2.13.5.3.5.	
ScrollTableEntryToRow	See Section 2.22.1.34.2.	

Table 2.13-119: ReconstFetch Information.

2.13.5.3.5 ReconstElementSelect

ReconstElementSelect is called to build the reconst vehicle table for the selected element in the reconst element table. It checks for validity of the selection and brings up the next dialog describing the selection. When this function is called *box* and*theEvent* may be null. The function call is ReconstElementSelect(defn, row, box, theEvent). Table 2.13-120 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
element	ReconstElementHandle	Development:SIMNET:MCC: SCC:reconscr.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
t1	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
t2	register *ScrollTableEntryHandle*	Development:SIMNET:libmac: scroll.h
i	register int	Standard C type.
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
vehicle	ReconstVehicleHandle	Development:SIMNET:MCC: SCC:reconscr.h
first	int	Standard C type.
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
ScrollTableRowToEntry	See Section 2.22.1.34.1.	
SetScrollTableSelection	See Section 2.22.1.37.2.	
DisposHandle	Standard Memory Manager function for Macintosh.	
SetCtlMax	Standard Control Manager function for Macintosh.	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
NewReconstVehicle	See Section 2.13.5.2.2.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
Called By		
Function	**Where Described**	
ReconstFetch	See Section 2.13.5.3.4.	

Table 2.13-120: ReconstElementSelect Information.

2.13.5.3.6 ReconstVehicleSelect

ReconstVehicleSelect is called when an entry in the reconst vehicle table is selected. It checks for validity of selection and brings up the next dialog describing the selection. The function call is ReconstVehicleSelect(defn, row, box, theEvent). Table 2.13-121 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
Calls		
Function	**Where Described**	
ScrollTableRowToEntry	See Section 2.22.1.34.1.	
SetScrollTableSelection	See Section 2.22.1.37.2.	

Table 2.13-121: ReconstVehicleSelect Information.

2.13.5.3.7 ReconstEvent

ReconstEvent is called when a button in the reconst dialog is pressed. It determines whether the data entered is valid and puts up the appropriate dialog. The routine returns the item number of the button pressed. The function call is ReconstEvent(dialog, itemNo). Table 2.13-122 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of the button hit

Calls	
Function	**Where Described**
ShowVehicleDialog	See Section 2.22.2.9.5.
HideWindow	Standard Window Manager function for Macintosh.
ShowWindow	Standard Window Manager function for Macintosh.
DisposeScrollTable	See Section 2.22.1.30.2.
DialogSegNext	See Section 2.22.1.39.4.
TruckReconstStart	See Section 2.13.5.3.2.
ShowBtryReconstDialog	See Section 2.13.4.2.10.
ShowCEWReconstDialog	See Section 2.13.4.4.10.

Table 2.13-122: ReconstEvent Information.

2.13.6 Simulating Gunnery Targets

2.13.6.1 target.h

Development:SIMNET:MCC:SCC:target.h

target.h is the header file associated with the following three files; target.c, tgtentry.c and tgttable.c. Table 2.13-123 describes the variables used by target.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
targetTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
targetColumns	extern array of ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
targetTableDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
targetDialog	poiner to extern DialogState	Development:SIMNET:libmac:dialog.h

Table 2.13-123: target.h Variable Information.

2.13.6.2 target.c

Development:SIMNET:MCC:SCC:target.c

target.c contains routines for creating, discarding, and communicating to the host descriptions of gunnery targets. Table 2.13-124 describes the variables used by target.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
targetDialog	poiner to DialogState	Development:SIMNET:libmac:dialog.h
targetByNumber	array of numberGunneryTargets TargetHandle	Development:SIMNET:MCC:SCC:target.h
numberExistingTargets	int	Standard C type.

Table 2.13-124: target.c Variable Information.

2.13.6.2.1 NewTarget

NewTarget allocates a new target descriptor. The function call is NewTarget(). Table 2.13-125 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tgt	register TargetHandle	Development:SIMNET:MCC:SCC:target.h
i	short	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
tgt	TargetHandle	Allocated target descriptor.
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
DisableControl	See Section 2.22.1.7.1.	
Called By		
Function	**Where Described**	
TargetTableEvent	See Section 2.13.6.4.9.	
ReadGunneryTargets	See Section 2.13.8.12.4.	

Table 2.13-125: NewTarget Information.

2.13.6.2.2 RemoveTarget

RemoveTarget throws away a target, *tgt*. The host is notified that the target does not exist any more, the scrolling table of targets is updated, and the New Target button is enabled if the maximum number of targets has not been reached. The function call is RemoveTarget(tgt). Table 2.13-126 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
tgt	register TargetHandle	Development:SIMNET:MCC:SCC:target.h
Calls		
Function	**Where Described**	
UploadTarget	See Section 2.13.6.2.5.	
RemoveScrollTableEntry	See Section 2.22.1.28.1.	
EnableControl	See Section 2.22.1.7.2.	
Called By		
Function	**Where Described**	
TargetEntryEvent	See Section 2.13.6.3.3.	
WipeTargetTable	See Section 2.13.8.12.5.	

Table 2.13-126: RemoveTarget Information.

2.13.6.2.3 TargetHit

TargetHit is called when the host has notified us that the target has been destroyed. The function call is TargetHit(n). Table 2.13-127 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
n	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tgt	register TargetHandle	Development:SIMNET:MCC:SCC:target.h
Calls		
Function	**Where Described**	
UpdateScrollTableEntry	See Section 2.22.1.31.1.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.13.8.6.1.	

Table 2.13-127: TargetHit Information.

2.13.6.2.4 ResetTargets

ResetTargets restores destroyed targets to health. The function call is ResetTargets(). Table 2.13-128 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	short	Standard C type.
tgt	register TargetHandle	Development:SIMNET:MCC:SCC:target.h
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
UpdateScrollTableEntry	See Section 2.22.1.31.1.	
UploadTarget	See Section 2.13.6.2.5.	
Called By		
Function	**Where Described**	
TargetTableEvent	See Section 2.13.6.4.9.	

Table 2.13-128: ResetTargets Information.

2.13.6.2.5 UploadTarget

UploadTarget notifies the host about a gunnery target, *tgt*. This function will only compile if VERSION is APPLETALK. Otherwise, it is a dummy function. The function call is UploadTarget(tgt). Table 2.13-129 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
tgt	TargetHandle	Development:SIMNET:MCC:SCC:target.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCTargetSetRequest	Development:SIMNET:MCC:SCC:SCC.h
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
RemoveTarget	See Section 22.13.6.2.2	
ResetTargets	See Section 2.13.6.2.4.	
TargetEntryEvent	See Section 2.13.6.3.3.	
ReadGunneryTargets	See Section 2.13.8.12.4.	

Table 2.13-129: UploadTarget Information.

2.13.6.3 tgtentry.c
Development:SIMNET:MCC:SCC:tgtentry.c

tgtentry.c implements the dialog for defining a gunnery target. Table 2.13-130 describes the variables used by tgtentry.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
targetEntryDialogDefn	extern DialogDefn	Development:SIMNET:libmac: dialog.h
currentTarget	TargetHandle	Development:SIMNET:MCC: SCC:target.h
targetBuffer	GunneryTargetDescriptor	Development:SIMNET:MCC: SCC:target.h
targetCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
targetRemoveField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
targetNameField	TextFieldDefn	Development:SIMNET:libmac: dialog.h
targetTypeFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
targetForceFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
targetLocationField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
targetAzimuthField	NumberFieldDefn	Development:SIMNET:libmac: dialog.h
targetEntryFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
targetEntryDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.13-130: tgtentry.c Variable Information.

2.13.6.3.1 ShowTarget

ShowTarget pops up a dialog which allows the definition of a gunnery target, *tgt*. The function call is ShowTarget(tgt, box). Table 2.13-131 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
tgt	TargetHandle	Development:SIMNET:MCC: SCC:target.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Calls	
Function	**Where Described**
ShowDialog	See Section 2.22.1.11.1.
LabelForceButtons	See Section 2.22.1.16.1.
ZoomToWindow	See Section 2.22.1.47.4.
ShowWindow	Standard Window Manager function for Macintosh.

Called By	
Function	**Where Described**
TargetTableSelect	See Section 2.13.6.4.3.
TargetTableEvent	See Section 2.13.6.4.9.

Table 2.13-131: ShowTarget Information.

2.13.6.3.2 TargetEntryFetch

TargetEntryFetch reads the current values of the target entries into the dialog fields for display to the user. The function call is TargetEntryFetch(state). Table 2.13-132 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Called By		
Function	**Where Described**	
TargetEntryEvent	See Section 2.13.6.3.3.	

Table 2.13-132: TargetEntryFetch Information.

2.13.6.3.3 TargetEntryEvent

TargetEntryEvent is called when a button in the target entry dialog is pressed. If the Help button is pressed, the Help dialog is put up. If the Cancel button is pressed, the dialog is updated. If the Remove button is pressed, the selected target is removed. If the OK button is pressed, the validity of the selection is checked, and the entry dialog is put up. The item number of the button pressed is returned. The function call is TargetEntryEvent(state, itemNo). Table 2.13-133 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
row	int	Standard C type.
box	Rect	Development:THINK C: Mac #includes:MacTypes.h

Calls	
Function	**Where Described**
showhelp	See Section 2.22.1.19.4.
TargetEntryFetch	See Section 2.13.6.3.2.
UpdateDialog	See Section 2.22.1.11.2.
RemoveTarget	See Section 2.13.6.2.2.
ThrowDialog	See Section 2.22.1.11.3.
CheckMandatoryFields	See Section 2.22.1.13.1.
InstallScrollTableEntry	See Section 2.22.1.21.
ScrollTableEntryToRow	See Section 2.22.1.34.2.
ScrollTableRowRect	See Section 2.22.1.34.3.
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
UploadTarget	See Section 2.13.6.2.5.

Table 2.13-133: TargetEntryEvent Information.

2.13.6.4 tgttable.c

Development:SIMNET:MCC:SCC:tgttable.c

tgttable.c implements the dialog displaying the scrolling table of gunnery targets. Table 2.13-134 describes the variables used by tgttable.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
targetColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
targetTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
targetTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
targetTableDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.13-134: tgttable.c Variable Information.

2.13.6.4.1 SetUpTargets

SetUpTargets initializes the target table. The Target Table dialog is opened and the Target List is created. The function call is SetUpTargets(). Table 2.13-135 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
CharWidth	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
ShowDialog	See Section 2.22.1.11.1.

Called By	
Function	**Where Described**
SetUp	See Section 2.13.8.17.1.

Table 2.13-135: SetUpTargets Information.

2.13.6.4.2 ShowTargetTable

ShowTargetTable makes the gunnery target table dialog visible. The function call is ShowTargetTable(). Table 2.13-136 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
SelectWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
BMOverviewEvent	See Section 2.13.8.16.4.

Table 2.13-136: ShowTargetTable Information.

2.13.6.4.3 TargetTableSelect

TargetTableSelect is called when an entry in the target table is selected. It checks for validity of the selection and brings up the next dialog describing the selection. The function call is TargetTableSelect(defn, row, box). Table 2.13-137 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
Calls		
Function	**Where Described**	
ScrollTableRowToEntry	See Section 2.22.1.34.1.	
ShowTarget	See Section 2.13.6.3.1.	

Table 2.13-137: TargetTableSelect Information.

2.13.6.4.4 TargetTableHilite

TargetTableHilite draws a box around the given selection so the user knows the item is selected. The function call is TargetTableHilite(defn, entry, box). Table 2.13-138 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hitPattern	Pattern	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
PenMode	Standard Quickdraw function for Macintosh.	
PenPat	Standard Quickdraw function for Macintosh.	
PaintRect	Standard Quickdraw function for Macintosh.	
PenNormal	Standard Quickdraw function for Macintosh.	

Table 2.13-138: TargetTableHilite Information.

2.13.6.4.5 TargetDrawName

TargetDrawName fills in the target name for the specified column in the target table to display to the user. The function call is TargetDrawName(defn, col, entry, box). Table 2.13-139 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.13-139: TargetDrawName Information.

2.13.6.4.6 TargetDrawType

TargetDrawType fills in the type field for the specified column in the target table according to the data structures to display to the user. The function call is TargetDrawType(defn, col, entry, box). Table 2.13-140 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
typeString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-140: TargetDrawType Information.

2.13.6.4.7 TargetDrawAppearance

TargetDrawAppearance fills in the appearance field for the specified column in the target table to display to the user. The function call is TargetDrawAppearance(defn, col, entry, box). Table 2.13-141 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-141: TargetDrawAppearance Information.

2.13.6.4.8 TargetDrawLocation

TargetDrawLocation fills in the location field for the specified column in the target table to display to the user. The function call is TargetDrawLocation(defn, col, entry, box). Table 2.13-142 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.13-142: TargetDrawLocation Information.

2.13.6.4.9 TargetTableEvent

TargetTableEvent is called when an event occurs in the target table dialog. If the Help button is pressed, the help dialog is put up. If the New button is pressed, it gets a new target. If the Overview button is pressed, the overview screen is put up. The item number of the button pressed is returned. The function call is TargetTableEvent(dialog, itemNo). Table 2.13-143 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
tgt	register TargetHandle	Development:SIMNET:MCC: SCC:truck.h
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h

Calls	
Function	**Where Described**
showhelp	See Section 2.22.1.19.4.
NewTarget	See Section 2.13.6.2.1.
GetDItem	Standard Dialog Manager function for Macintosh.
ShowTarget	See Section 2.13.6.3.1.
ResetTargets	See Section 2.13.6.2.4.
HideWindow	Standard Window Manager function for Macintosh.

Table 2.13-143: TargetTableEvent Information.

2.13.7 Visual Appearance of SCC User Interface

2.13.7.1 SCC Pictures
Development:SIMNET:MCC:SCC:SCC Pictures

SCC Pictures contains resources that determine the appearance of the SCC application's user interface.

2.13.7.2 resource.h
Development:SIMNET:MCC:SCC:resource.h

resource.h defines the numbers of the resources present in the SCC Pictures resource file.

2.13.8 Auxiliary Software

2.13.8.1 SCC.h
Development:SIMNET:MCC:SCC:SCC.h

SCC.h defines the representation of information communicated between the SCC Macintosh application and the MCC host.

2.13.8.2 SCCMac.h
Development:SIMNET:MCC:SCC:SCCMac.h

SCCMac.h defines types, constants, and routines used within the SCC Macintosh application. Table 2.13-144 describes the variable used by SCCMac.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
battleScheme	extern char	Standard C type.
forceID	extern char	Standard C type.
terrainSelection	extern char	Standard C type.
terrains	extern array of TerrainSelection	Development:SIMNET:MCC:SCC:SCCMac.h
optionalElementInfo	extern array of OptionalElementInfo	Development:SIMNET:MCC:SCC:SCCMac.h
optionalElements	extern OptionalElements	Development:SIMNET:MCC:include:options.h
numberHowitzerBatteries	extern int	Standard C type.
trainsOrganization	extern char	Standard C type.
supportPltLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
fuelLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
ammoLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
umcpLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
coTrainLocations	extern array of maxNumberCompanies MapCoordinates	Development:SIMNET:libmac:map.h
class3SPLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
class3DPLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
aspLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
atpLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
adminLogLocation	extern MapCoordinates	Development:SIMNET:libmac:map.h
ammoSupplyRate	extern array of numberAmmoVarieties int	Standard C type.
seqDialog	pointer to DialogSeqState	Development:SIMNET:libmac:sequence.h
entryDialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
initOverviewDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
bmOverviewDailog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
watchCursorHandle	extern CursHandle	Development:THINK C:Mac #includes:Quickdraw.h
bmPassword	extern array of char	Standard C type.

Table 2.13-144: SCCMac.h Variable Information.

2.13.8.3 SCC #includes.c

Development:SIMNET:MCC:SCC:SCC #includes.c

SCC #includes.c is compiled by the THINK C compiler to produce a precompiled header file, "SCC Headers", that is included by various SCC source code files.

2.13.8.4 SCCoptions.h
Development:SIMNET:MCC:SCC:SCCoptions.h

SCCoptions.h associates a constant with each type of optional element that may be simulated by the MCC system.

2.13.8.5 version.h
Development:SIMNET:MCC:SCC:version.h

version.h defines constants that determine which version of the SCC application is compiled.

2.13.8.6 atalk.c
Development:SIMNET:MCC:SCC:atalk.c

atalk.c contains routines supporting AppleTalk communication between the SCC Macintosh application and the MCC host. The functions in this file are only compiled if VERSION is APPLETALK.

2.13.8.6.1 ProcessRequest

ProcessRequest processes a request received from the MCC host. The function call is ProcessRequest(hdl). Table 2.13-145 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
hdl	Handle	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	register Ptr	Development:THINK C: Mac #includes:MacTypes.h
errCode	int	Standard C type.
Calls		
Function	**Where Described**	
SetDateTime	Standard Operating System Utility function for Macintosh.	
TargetHit	See Section 2.13.6.2.3.	
ATPResponse	Standard Appletalk Manager function for Macintosh.	
Restart	Standard Operating System Utility function for Macintosh.	
ProcessSimPlacedRequest	See Section 2.22.2.3.4.	
SimulatorPlaced	See Section 2.13.2.2.8.	
ProcessSimProblemRequest	See Section 2.13.8.6.2.	
ATPError	See Section 2.22.1.3.6.	
Called By		
Function	**Where Described**	
main	See Section 2.13.8.15.1.	

Table 2.13-145: ProcessRequest Information.

2.13.8.6.2 ProcessSimProblemRequest

ProcessSimProblemRequest reports a problem, *problem*, with a vehicle simulator, *vehicle*. The function call is ProcessSimProblemRequest(vehicle, problem). Table 2.13-146 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	short	Standard C type.
problem	char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cStr	array of 100 char	Standard C type.
Calls		
Function	**Where Described**	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.13.8.6.1.	

Table 2.13-146: ProcessSimProblemRequest Information.

2.13.8.6.3 DownloadTerrain

DownloadTerrain obtains information about terrain patches from the host. The function call is DownloadTerrain(). Table 2.13-147 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
tp	register pointer to TerrainSelection	Development:SIMNET:MCC: SCC:SCCMac.h
ab1	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
ab2	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
req1	SCCTerrainRequest	Development:SIMNET:MCC: SCC:SCC.h
rsp1	SCCTerrainResponse	Development:SIMNET:MCC: SCC:SCC.h
req2	SCCMapSheetsRequest	Development:SIMNET:MCC: SCC:SCC.h
rsp2	SCCMapSheetsResponse	Development:SIMNET:MCC: SCC:SCC.h
errCode	int	Standard C type.

Calls	
Function	**Where Described**
NewHandle	Standard Memory Manager function for Macintosh.
SetUpATPRequest	See Section 2.22.1.3.2.
ATPError	See Section 2.22.1.3.6.
ATPRequest	Standard Appletalk Manager function for Macintosh.
DisposHandle	Standard Memory Manager function for Macintosh.
Called By	
Function	**Where Described**
main	See Section 2.13.8.15.1.

Table 2.13-147: DownloadTerrain Information.

2.13.8.6.4 DownloadOptions

DownloadOptions obtains information about which optional elements are supportable. The function call is DownloadOptions(). Table 2.13-148 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
rsp	SCCPermitOptionsResponse	Development:SIMNET:MCC:SCC:SCC.h
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	
PermitOptionalElements	See Section 2.13.8.11.9.	
Called By		
Function	**Where Described**	
main	See Section 2.13.8.15.1.	

Table 2.13-148: DownloadOptions Information.

2.13.8.7 data.c
Development:SIMNET:MCC:SCC:data.c

data.c defines various data structures used by the SCC application, such as tables of information about supply trucks. Table 2.13-149 describes the variables used by data.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
application	pointer to char	Standard C type.
authors	pointer to char	Standard C type.
copyright	pointer to char	Standard C type.
terrainSelection	char	Standard C type.
terrains	array of maxNumberTerrains TerrainSelection	Development:SIMNET:MCC: SCC:SCCMac.h
optionalElements	OptionalElements	Development:SIMNET:MCC: include:options.h
numberHowitzerBatteries	int	Standard C type.
trainsOrganization	char	Standard C type.
supportPltLocation	MapCoordinates	Development:SIMNET:libmac: map.h
fuelLocation	MapCoordinates	Development:SIMNET:libmac: map.h
ammoLocation	MapCoordinates	Development:SIMNET:libmac: map.h
umcpLocation	MapCoordinates	Development:SIMNET:libmac: map.h
coTrainLocations	array of maxNumbercompanies MapCoordinates	Development:SIMNET:libmac: map.h
class3SPLocation	MapCoordinates	Development:SIMNET:libmac: map.h
class3DPLocation	MapCoordinates	Development:SIMNET:libmac: map.h
aspLocation	MapCoordinates	Development:SIMNET:libmac: map.h
atpLocation	MapCoordinates	Development:SIMNET:libmac: map.h
ammoSupplyRate	array of numberAmmoVarieties int	Standard C type.
seqDialog	pointer to DialogSeqState	Development:SIMNET:libmac: sequence.h
entryDialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
watchCursorHandle	CursHandle	Development:THINK C: Mac #includes:Quickdraw.h
bmPassword	array of bmPasswordLength char	Standard C type.

Table 2.13-149: data.c Variable Information.

2.13.8.8 dialog.c
Development:SIMNET:MCC:SCC:dialog.c

dialog.c contains miscellaneous routines supporting dialog implementations.

2.13.8.8.1 CheckMandatoryEvent

CheckMandatoryEvent checks to see if the button pressed was the Next button and checks the validity of the next field. The function call is CheckMandatoryEvent(dialog, itemNo). Table 2.13-150 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	

Table 2.13-150: CheckMandatoryEvent Information.

2.13.8.8.2 ZoomUpFromTableEntry

ZoomUpFromTableEntry draws an animated rectangle that starts the size of the table entry and shrinks to the size of the pop-up window, then displays the pop-up window. The function call is ZoomUpFromTableEntry(box). Table 2.13-151 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
box	register pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
AllocTableSelect	See Section 2.13.2.1.3.	
PlaceTableSelect	See Section 2.13.2.2.6.	
TruckTableSelect	See Section 2.13.3.4.10.	

Table 2.13-151: ZoomUpFromTableEntry Information.

2.13.8.8.3 ZoomDownToTableEntry

ZoomDownToTableEntry hides the pop-up window and then draws an animated rectangle that grows from the size of the pop-up window to the size of the table entry. The function call is ZoomDownToTableEntry(). Table 2.13-152 describes the functions called using this function.

Calls	
Function	**Where Described**
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
Called By	
Function	**Where Described**
AllocEntryEvent	See Section 2.13.2.1.6.
PlaceComplete	See Section 2.13.2.2.5.
TruckTableComplete	See Section 2.13.3.4.9.

Table 2.13-152: ZoomDownToTableEntry Information.

2.13.8.9 displace.h
Development:SIMNET:MCC:SCC:displace.h

displace.h header file associated with the following file, displace.c. Table 2.13-153 describes the variables used by displace.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
displaceTable	extern array of DisplaceData	Development:SIMNET:MCC:SCC:displace.h

Table 2.13-153: displace.h Variable Information.

2.13.8.10 displace.c
Development:SIMNET:MCC:SCC:displace.c

displace.c implements the displacement of units such as howitzer batteries and command posts. Table 2.13-154 describes the variables used by displace.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
dispElementColumn	ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
dispElementTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
dispUnitColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
dispUnitTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
displaceFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
displaceDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
dispElementSelection	int	Standard C type.
dispUnitSelection	int	Standard C type.
newLocation	MapCoordinates	Development:SIMNET:libmac: map.h
displaceTable	array of displaceTableMaxEntries DisplaceData	Development:SIMNET:MCC: SCC:displace.h
supplyDepotsInited	extern char	Standard C type.
displaceDispatchDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
displaceHaltDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
bmOverviewDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
displaceDispatchButtonFields	array of PushButtonfieldDefn	Development:SIMNET:libmac: dialog.h
displaceDispatchNewLocation Field	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
displaceDispatchFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
displaceDispatchDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
displaceHaltButtonFields	array of PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
displaceHaltFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
displaceHaltDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-154: displace.c Variable Information.

2.13.8.10.1 CompleteDisplacement

CompleteDisplacement is called when the displacement dialog should be checked for correctness and taken down. The function call is CompleteDisplacement(). Table 2.13-155 describes the functions which call this function.

Called By	
Function	**Where Described**
ProcessCommandShiftOption Sequence	See Section 2.13.8.15.4.

Table 2.13-155: CompleteDisplacement Information.

2.13.8.10.2 DisplaceFetch

DisplaceFetch reads the current values of the howitzers, TOC, ALOC, UMCP, and depot elements into the displacement dialog fields to display to the user. The function call is DisplaceFetch(state). Table 2.13-156 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
element	register DispElementHandle	Development:SIMNET:MCC: SCC:displace.h
Calls		
Function	**Where Described**	
NewDispElement	See Section 2.13.8.10.9.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
DisplaceSelectElement	See Section 2.13.8.10.5.	
ScrollTableEntryToRow	See Section 2.22.1.34.2.	

Table 2.13-156: DisplaceFetch Information.

2.13.8.10.3 DisplaceEvent

DisplaceEvent checks the status of the Displace Table. If the status is stationary, the Dispatch dialog is put up. Otherwise, the Halt dialog is put up. The function call is DisplaceEvent(dialog, itemNo). Table 2.13-157 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of the button hit

Table 2.13-157: DisplaceEvent Information.

2.13.8.10.4 DisplaceBranch

DisplaceBranch is used when there is a list of items to select. It checks the type of item selected and brings up the next dialog, depending on the item type. The function call is DisplaceBranch(state). Table 2.13-158 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogSeqState	Development:SIMNET:libmac: sequence.h
Return Values		
Return Value	**Type**	**Meaning**
&displaceDispatchDialog	pointer to DialogNodeDefn	handle of the displace Dispatch dialog
&displaceHaltDialog	pointer to DialogNodeDefn	handle of the displace Halt dialog

Table 2.13-158 DisplaceBranch Information.

2.13.8.10.5 DisplaceSelectElement

DisplaceSelectElement is called when an element entry in the dispatch table is selected. It checks for validity of selection and brings up the next dialog describing the selection. The function call is DisplaceSelectElement(defn, row, box, theEvent). Table 2.13-159 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
t1	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
t2	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
unit	register DispUnitHandle	Dvelopment:SIMNET:MCC: SCC:displace.h
numHalves	int	Standard C type.
i	int	Standard C type.
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
numberHowitzerBatteries	extern int	Standard C type.
Calls		
Function	**Where Described**	
ScrollTableRowToEntry	See Section 2.22.1.34.1.	
SetScrollTableSelection	See Section 2.22.1.37.2.	
DisposHandle	Standard Memory Manager function for Macintosh.	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
NewDispUnit	See Section 2.13.8.10.8.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
Called By		
Function	**Where Described**	
DisplaceFetch	See Section 2.13.8.10.2.	

Table 2.13-159: DisplaceSelectElement Information.

2.13.8.10.6 DisplaceSelectUnit

DisplaceSelectUnit is called when a unit entry in the dispatch table is selected. It checks for the validity of selection and brings up the next dialog describing the selection. The function call is DisplaceSelectUnit(defn,row, box, theEvent). Table 2.13-160 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
Calls		
Function	**Where Described**	
SetScrollTableSelection	See Section 2.22.1.37.2.	
ScrollTableRowToEntry	See Section 2.22.1.34.1.	

Table 2.13-160: DisplaceSelectUnit Information.

2.13.8.10.7 NewDispElement

NewDispElement allocates memory for a new element in the displacement table and sets up fields. The function call is NewDispElement(key). Table 2.13-161 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
key	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
element	register DispElementHandle	Development:SIMNET:MCC: SCC:displace.h
Return Values		
Return Value	**Type**	**Meaning**
element	DispElementHandle	Handle to the data structure describing the entry that was just created
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	

Table 2.13-161: NewDispElement Information.

2.13.8.10.8 NewDispUnit

NewDispUnit allocates memory for a new unit in the displacement table. The function call is NewDispUnit(index). Table 2.13-162 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
index	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
unit	register DispUnitHandle	Development:SIMNET:MCC:SCC:displace.h
Return Values		
Return Value	**Type**	**Meaning**
unit	DispUnitHandle	Handle to the data structure describing the entry that was just created
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
Called By		
Function	**Where Described**	
DisplaceSelectElement	See Section 2.13.8.10.5.	

Table 2.13-162: NewDispUnit Information.

2.13.8.10.9 DrawDispElement

DrawDispElement fills in element fields of the specified column in the displace table to display to the user. The function call is DrawDispElement(defn, col, entry, box). Table 2.13-163 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac:scroll.h
box	pointer to Rect	Development:THINK C:Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
Move	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	
Called By		
Function	**Where Described**	
DisplaceFetch	See Section 2.13.8.10.2.	

Table 2.13-163: DrawDispElement Information.

2.13.8.10.10 DrawDispUnit

DrawDispUnit fills in unit fields of the specified column in the displace table to display to the user. The function call is DrawDispUnit(defn, col, entry, box). Table 2.13-164 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
index	register int	Standard C type.
cStr	array of 30 char	Standard C type.
pStr	array of 30 char	Standard C type.
unitNo	short	Standard C type.
half	char	Standard C type.
Calls		
Function	**Where Described**	
Move	*Standard Quickdraw function for Macintosh.*	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-164: DrawDispUnit Information.

2.13.8.10.11 DrawDispStatus

DrawDispStatus fills in status fields of the specified column in the table to display to the user. The function call is DrawDispStatus(defn, col, entry, box). Table 2.13-165 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
unit	register DispUnitHandle	Development:SIMNET:MCC:SCC:displace.h
index	register int	Standard C type.
cStr	array of 30 char	Standard C type.
pStr	array of 30 char	Standard C type.
Calls		
Function	**Where Described**	
Move	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-165: DrawDispStatus Information.

2.13.8.10.12 DrawDispETA

DrawDispETA fills in ETA fields of the specified column in the displace table according to the data structures to display to the user. The function call is DrawDispETA(defn, col, entry, box). Table 2.13-166 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac:scroll.h
box	pointer to Rect	Development:THINK C:Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
unit	register DispUnitHandle	Development:SIMNET:MCC:SCC:displace.h
index	register int	Standard C type.
cStr	array of 30 char	Standard C type.
pStr	array of 30 char	Standard C type.
Calls		
Function	**Where Described**	
DTGToString	See Section 2.22.1.15.1.	
Move	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.13-166: DrawDispETA Information.

2.13.8.10.13 SetUpDisplacement

SetUpDisplacement sets up the displacment dialog. The function call is SetUpDisplacement(). Table 2.13-167 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
Called By	
Function	**Where Described**
SetUp	See Section 2.13.8.17.1.

Table 2.13-167: SetUpDisplacement Information.

2.13.8.10.14 DisplaceDispatchFetch

DisplaceDispatchFetch reads the current values of the dispatch unit data from the displace table into the displace dispatch dialog fields to display to the user. The function call is DisplaceDispatchFetch(state). Table 2.13-168 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pLocation	array of 30 char	Standard C type.
cDispUnit	array of 30 char	Standard C type.
pDispUnit	array of 30 char	Standard C type.
data	register pointer to DisplaceData	Development:SIMNET:MCC:SCC:displace.h
Calls		
Function	**Where Described**	
ParamText	Standard Dialog Manager function for Macintosh.	

Table 2.13-168: DisplaceDispatchFetch Information.

2.13.8.10.15 DisplaceDispatchEvent

DisplaceDispatchEvent is called when a button in the displace dispatch dialog is pressed. It determines whether the data entered is valid. It throws the current dialog away and goes to the main screen. It returns the item number of the button pressed. The function call is DisplaceDispatchEvent(dialog, itemNo). Table 2.13-169 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
data	register pointer to DisplaceData	Development:SIMNET:MCC:SCC:displace.h
etaString	array of 30 char	Standard C type.
tempTimeEntroute	unsigned long	Standard C type.
tempETA	DateTimeGroup	Development:SIMNET:libmac:dtg.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of the pressed button
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
NetworkDispatched	See Section 2.13.8.10.21.	
ComputeETA	See Section 2.13.8.10.18.	
Get_DTG	See Section 2.22.1.15.4.	
DTGToString	See Section 2.22.1.15.4.	
SetText	See Section 2.22.1.11.8.	

Table 2.13-169: DisplaceDispatchEvent Information.

2.13.8.10.16 DisplaceHaltFetch

DisplaceHaltFetch reads the current values of the displace data structures into the displace halt dialog fields to display to the user. The function call is DisplaceHaltFetch(state). Table 2.13-170 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
data	register pointer to DisplaceData	Development:SIMNET:MCC:SCC:displace.h
cDispUnit	array of 30 char	Standard C type.
pDispUnit	array of 30 char	Standard C type.
pLocation	array of 30 char	Standard C type.
pETA	array of 30 char	Standard C type.

Calls	
Function	**Where Described**
DTGToString	See Section 2.22.1.15.1.
ParamText	Standard Dialog Manager function for Macintosh.

Table 2.13-170: DisplaceHaltFetch Information.

2.13.8.10.17 DisplaceHaltEvent

DisplaceHaltEvent is called when a button in the displace halt dialog is pressed. It determines whether the data entered is valid. It throws the current dialog away and goes to the main screen. It returns the item number of the button pressed. The function call is DisplaceHaltEvent(dialog, itemNo). Table 2.13-171 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
data	register pointer to DisplaceData	Development:SIMNET:MCC:SCC:displace.h
percent	unsigned long	Standard C type.
time	DateTimeGroup	Development:SIMNET:libmac:dtg.h
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of pressed button
Calls		
Function	**Where Described**	
Get_DTG	See Section 2.22.1.15.4.	
comp_pct	See Section 2.13.8.10.19.	
PercentPt	See Section 2.22.1.23.3.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
NetworkArrived	See Section 2.13.8.10.22.	

Table 2.13-171: DisplaceHaltEvent Information.

2.13.8.10.18 ComputeETA

ComputeETA computes an ETA given the current location, *Location*, destination, *Destination*, and speed, *Speed*. It returns the ETA in *eta*, and the total travel time in *timeEnroute*. The function call is ComputeETA(Location, Destination, eta, timeEnroute, speed). Table 2.13-172 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
Location	MapCoordinates	Development:SIMNET:libmac:map.h
Destination	MapCoordinates	Development:SIMNET:libmac:map.h
eta	pointer to DateTimeGroup	Development:SIMNET:libmac:dtg.h
timeEnroute	pointer to unsigned long	Standard C type.
speed	short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
time	unsigned long	Standard C type.
triptime	unsigned long	Standard C type.
distance	unsigned long	Standard C type.
Calls		
Function	**Where Described**	
StringToMapCoordinates	See Section 2.22.1.26.1.	
DistBetween2Pts	See Section 2.22.1.23.2.	
GetDateTime	Standard Operating System Utility function for Macintosh.	
Get_DTG	See Section 2.22.1.15.4.	
DTGElapsed	See Section 2.22.1.15.3.	
Called By		
Function	**Where Described**	
DisplaceDispatchEvent	See Section 2.13.8.10.5.	

Table 2.13-172: ComputeETA Information.

2.13.8.10.19 comp_pct

comp_pct computes a percent difference between two times. The function call is comp_pct(time1, time2). Table 2.13-173 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
time1	unsigned long	Standard C type.
time2	unsigned long	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
percent	long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
(100 - percent)	unsigned long	percent difference between two times

Called By	
Function	**Where Described**
DisplaceHaltEvent	See Section 2.13.8.10.17.

Table 2.13-173: comp_pct Information.

2.13.8.10.20 UpdateDisplacement

UpdateDisplacement updates the displacement table entries. The function call is UpdateDisplacement(). Table 2.13-174 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
time	DateTimeGroup	Development:SIMNET:libmac:dtg.h
i	register int	Standard C type
data	pointer to DisplaceData	Development:SIMNET:MCC:SCC:displace.h
cautionString	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
Get_DTG	See Section 2.22.1.15.4.	
NetworkArrived	See Section 2.13.8.10.22.	
UpdateScrollTableEntry	See Section 2.22.1.31.1.	
LookupScrollTableEntry	See Section 2.22.1.24.1.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
MainEventLoop	See Section 2.13.8.15.3.	

Table 2.13-174: UpdateDisplacement Information.

2.13.8.10.21 NetworkDispatched

NetworkDispatched sends a message to the host indicating the user has requested a supply depot, a howitzer battery, or that one of the command posts has been displaced. This function will only compile if VERSION is APPLETALK. Otherwise, it is a dummy function. The function call is NetworkDispatched(data). Table 2.13-175 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
data	register pointer to DisplaceData	Development:SIMNET:MCC:SCC:displace.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
artyReq	SCCArtyDispatchRequest	Development:SIMNET:MCC:SCC:SCC.h
depotsReq	SCCDepotsRequest	Development:SIMNET:MCC:SCC:SCC.h
umcpReq	SCCUMCPDisplaceRequest	Development:SIMNET:MCC:SCC:SCC.h
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
DisplaceDispatchEvent	See Section 2.13.8.10.15.	

Table 2.13-175: NetworkDispatched Information.

2.13.8.10.22 NetworkArrived

NetworkArrived is called when the user halts the displacement or the ETA has come about. It sends a message to the host indicating the new location of the unit. This function will only compile if VERSION is APPLETALK. Otherwise, it is a dummy function. The function call is NetworkArrived(data). Table 2.13-176 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
data	pointer to DisplaceData	Development:SIMNET:MCC:SCC:displace.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ab	ABRecHandle	
artyReq	pointer to SCCArtyArriveRequest	Development:SIMNET:MCC:SCC:SCC.h
TOCReq	pointer to SCCTOCRequest	Development:SIMNET:MCC:SCC:SCC.h
ALOCReq	pointer to SCCALOCRequest	Development:SIMNET:MCC:SCC:SCC.h
alocRole	extern char	Standard C type.
tocRole	extern char	Standard C type.
tocConfiguration	extern char	Standard C type.
GetGunAzimuth	extern array of int	See 2.13.4.2.12.
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
NewPtr	Standard Memory Manager function for Macintosh.	
GetGunAzimuth	See Section 2.13.4.2.12.	
SetUpATPRequest	See Section 2.22.1.3.2.	
ATPRequest	Standard Appletalk Manager function for Macintosh.	

Called By	
Function	**Where Described**
DisplaceHaltEvent	See Section 2.13.8.10.17.
UpdateDisplacement	See Section 2.13.8.10.20.

Table 2.13-176: NetworkArrived Information.

2.13.8.11 exercise.c

Development:SIMNET:MCC:SCC:exercise.c

exercise.c implements the user interface for exercise initialization (i.e., choosing a terrain database, selecting participating optional elements, etc.). Table 2.13-177 describes the variables used by exercise.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
startDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
mccRoleDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
selectTerrainDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
optElement1Dialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
optElement2Dialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
coRoleDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
startConfirmDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
initOverviewDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
startFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
startDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
battleSchemeFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
mccForceFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
mccRoleFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
mccRoleDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
selectTerrainFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
selectTerrainFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
selectTerrainDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
elementsPermitted	OptionalElements	Development:SIMNET:MCC: include:options.h

(Table 2.13-177 is continued on the following page.)

Variable	Type	Where Typedef Declared
optElement1FieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
optElement1Dialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
optElement2Fields	array of CBFieldDefn	Development:SIMNET:libmac: dialog.h
optElement2HowBtrysField	NumberFieldDefn	Development:SIMNET:libmac: dialog.h
optElement2FieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
optElement2Dialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
coRoleFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
coRoleRectField	RectFieldDefn	Development:SIMNET:libmac: dialog.h
coRoleFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
coRoleDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
startConfirmfieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
startConfirmDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-177: exercise.c Variable Information.

2.13.8.11.1 LoadCannedExercise

LoadCannedExercise loads canned exercise initialization data. The function call is LoadCannedExercise(). Table 2.13-178 describes the functions which use this function.

Called By	
Function	**Where Described**
LoadCannedData	See Section 2.13.8.15.2.

Table 2.13-178: LoadCannedExercise Information.

2.13.8.11.2 MCCRoleFetch

MCCRoleFetch labels the force buttons in the MCC Role dialog. This dialog determines which force this MCC system is serving. The function call is MCCRoleFetch(dialog). Table 2.13-179 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h

Calls	
Function	**Where Described**
LabelForceButtons	See Section 2.22.1.16.1.

Table 2.13-179: MCCRoleFetch Information.

2.13.8.11.3 MCCRoleEvent

MCCRoleEvent is called when a battle scheme button in the MCC role dialog is changed. The labels on the force ID buttons are adjusted to match. The function call is MCCRoleEvent(dialog, itemNo). Table 2.13-180 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button
Calls		
Function	**Where Described**	
LabelForceButtons	See Section 2.22.1.16.1.	

Table 2.13-180: MCCRoleEvent Information.

2.13.8.11.4 SelectTerrainUpdate

SelectTerrainUpdate updates the terrain displayed for the current selection. The function call is SelectTerrainUpdate(dialog). Table 2.13-181 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
SetText	See Section 2.22.11.11.8.	
Called By		
Function	**Where Described**	
SelectTerrainFetch	See Section 2.13.8.11.5.	
SelectTerrainEvent	See Section 2.13.8.11.6.	

Table 2.13-181: SelectTerrainUpdate Information.

2.13.8.11.5 SelectTerrainFetch

SelectTerrainFetch reads the current names of the terrain databases into the dialog radio buttons for display to the user. The radio buttons are loaded with actual names of terrain databases. The exercise area corners and map sheets displayed are made to correspond to the current selection. The function call is SelectTerrainFetch(dialog). Table 2.13-182 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
j	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
str	array of 256 char	Standard C type.
Calls		
Function	**Where Described**	
GetDItem	Standard Dialog Manager function for Macintosh.	
SetCTitle	Standard Control Manager function for Macintosh.	
HideControl	Standard Control Manager function for Macintosh.	
SelectTerrainUpdate	See Section 2.13.8.11.4.	

Table 2.13-182: SelectTerrainFetch Information.

2.13.8.11.6 SelectTerrainEvent

SelectTerrainEvent is called when a button in the select terrain dialog is pressed. It determines whether the data entered is valid and updates the dialog. The item number of the button pressed is returned. The function call is SelectTerrainEvent(dialog, itemNo). Table 2.13-183 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button pressed

Calls	
Function	**Where Described**
SelectTerrainUpdate	See Section 2.13.8.11.4.

Table 2.13-183: SelectTerrainEvent Information.

2.13.8.11.7 LoadOptionsPermitted

This routine loads the permitted options for the selection optional elements 1 dialog. The function call is LoadOptionsPermitted(). Table 2.13-184 describes the functions which call this function.

Called By	
Function	**Where Described**
LoadCannedData	See Section 2.13.8.15.2.

Table 2.13-184 LoadOptionsPermitted Information.

2.13.8.11.8 SetUpOptionalElements

This routine sets up the optional elements. To start, all optional elements are selected (if permitted). The function call is SetUpOptionalElements(). Table 2.13-185 describes the functions which call this function.

Called By	
Function	**Where Described**
SetUp	See Section 2.13.8.17.1.

Table 2.13-185: SetUpOptionalElements Information.

2.13.8.11.9 PermitOptionalElements

PermitOptionalElements sets the optional elements permitted to *options*. The function call is PermitOptionalElements(options). Table 2.13-186 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
options	pointer to OptionalElements	Development:SIMNET:MCC: include:options.h
Called By		
Function	**Where Described**	
DownloadOptions	See Section 2.13.8.6.4.	

Table 2.13-186: PermitOptionalElements Information.

2.13.8.11.10 OptElement1Fetch

OptElement1Fetch reads the current values of the permitted options into the optional elements1 dialog fields to display to the user. This dialog permits the choice of whether to include TOC, Admin/Log Center, BN HQ Tank Section, Sct Plt, or Stealth Jeep. The function call is OptElement1Fetch(state). Table 2.13-187 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
DisableControl	See Section 2.22.1.7.1.	

Table 2.13-187: OptElement1Fetch Information.

2.13.8.11.11 OptElement2Fetch

OptElement2Fetch reads the current values of the permitted options into the optional elements 2 dialog fields to display to the user. This dialog permits the choice of whether to include Fire Support Element, Air Liaison Officer, or Combat Service Support. The function call is OptElement2Fetch(state). Table 2.13-188 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
state	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
theType	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
theRect	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
DisableControl	See Section 2.22.1.7.1.	

Table 2.13-188: OptElement2Fetch Information.

2.13.8.11.12 OptElement2Event

OptElement2Event is called when a button in the optional elements 2 dialog is pressed. It determines whether the data entered is valid. It throws the current dialog away and goes to the main screen. It returns the item number of the button pressed. The function call is OptElement2Event(dialog, itemNo). Table 2.13-189 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	*pointer* to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
str	array of 256 char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of button pressed
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
SelIText	Standard Dialog Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
SetText	See Section 2.22.1.11.8.	

Table 2.13-189: OptElement2Event Information.

2.13.8.11.13 CoRoleFetch

CoRoleFetch reads the current values of the company roles into the dialog fields to display to the user. The company role dialog allows the specification of each company's role in the exercise. The function call is CoRoleFetch(dialog). Table 2.13-190 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
DisableControl	See Section 2.22.1.7.1.	

Table 2.13-190: CoRoleFetch Information.

2.13.8.11.14 UploadExerciseParameters

UploadExerciseParameters uploads the current exercise parameters to the host. This function is only compiled if VERSION is APPLETALK. Otherwise, it is a dummy function.The function call is UploadExerciseParameters(). Table 2.13-191 describes the internal variables used, errors returned and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	SCCExerciseRequest	Development:SIMNET:MCC:SCC:SCC.h
rsp	SCCExerciseResponse	Development:SIMNET:MCC:SCC:SCC.h
errCode	int	Standard C type.
i	int	Standard C type.
Calls		
Function	**Where Described**	
SetCursor	Standard Quickdraw function for Macintosh.	
ATPPut	See Section 2.22.1.3.3.	
DownloadTerainMap	See Section 2.22.1.3.4.	
Called By		
Function	**Where Described**	
StartConfirmEvent	See Section 2.13.8.11.5.	

Table 2.13-191: UploadExerciseParameters Information.

2.13.8.11.15 StartConfirmEvent

The start confirm dialog solicits confirmation before starting an exercise. If the OK button was pressed, the exercise parameters are uploaded, and the next dialog is brought up. The item number of the button pressed is returned. The function call is StartConfirmEvent(dialog, itemNo). Table 2.13-192 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	the OK button was pressed
itemNo	int	resource id of button pressed
Calls		
Function	**Where Described**	
UploadExerciseParameters	See Section 2.13.8.11.14.	
DialogSeqNext	See Section 2.22.1.39.4.	

Table 2.13-192: StartConfirmEvent Information.

2.13.8.12 file.c

Development:SIMNET:MCC:SCC:file.c

file.c contains routines for creating and reading Macintosh files holding initialization parameters for gunnery targets and vehicle simulators. Table 2.13-193 describes the variables used by file.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
errStr	array of 128 char	Standard C type.

Table 2.13-193: file.c Variable Information

2.13.8.12.1 LoadGunneryTargets

LoadGunneryTargets loads the gunnery targets from a prepared file. The function call is LoadGunneryTargets(). Table 2.13-194 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
where	Point	Development:THINK C: Mac #includes:MacTypes.h
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	*FInfo*	Development:THINK C: Mac #includes:FileMgr.h
theFile	int	Standard C type.
theTypeList	SFTypeList	Development:THINK C: Mac #includes:StdFilePkg.h
count	long	Standard C type.
Calls		
Function	**Where Described**	
SFGetFile	Standard Standard File Package function for Macintosh.	
SetCursor	Standard Quickdraw function for Macintosh.	
FSOpen	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
WipeTargetTable	See Section 2.13.8.12.5.	
ReadGunneryTargets	See Section 2.13.8.12.4.	
FSClose	Standard File Manager function for Macintosh.	
FlushVol	Standard File Manager function for Macintosh.	
Called By		
Function	**Where Described**	
ProcessCommandShiftOption Sequence	See Section 2.13.8.15.4.	

Table 2.13-194: LoadGunneryTargets Information.

2.13.8.12.2 SaveGunneryTargets

SaveGunneryTargets saves the gunnery targets in a file selected by the user. The function call is SaveGunneryTargets(). Table 2.13-195 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
where	Point	Development:THINK C: Mac #includes:MacTypes.h
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	FInfo	Development:THINK C: Mac #includes:FileMgr.h
theFile	int	Standard C type.
count	long	Standard C type.
Calls		
Function	**Where Described**	
SFPutFile	Standard Standard File Package function for Macintosh.	
GetFInfo	Standard file Manager function for Macintosh.	
SetCursor	Standard Quickdraw function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
Create	Standard File Manager function for Macintosh.	
FSOpen	Standard File Manager function for Macintosh.	
WriteGunneryTargets	See Section 2.13.8.12.3.	
FSClose	Standard File Manager function for Macintosh.	
FlushVol	Standard file Manager function for Macintosh.	
Called By		
Function	**Where Described**	
ProcessCommandShiftOption Sequence	See Section 2.13.8.15.4.	

Table 2.13-195: SaveGunneryTargets Information.

2.13.8.12.3 WriteGunneryTargets

WriteGunneryTargets writes the gunnery target information in the file indicated by *theFile*. The function call is WriteGunneryTargets(theFile). Table 2.13-196 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
gt	TargetHandle	Development:SIMNET:MCC:SCC:target.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numTargets	int	Standard C type.
i	int	Standard C type.
bytesOut	int	Standard C type.
targetTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
FSWrite	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
SetEOF	Standard file Manager function for Macintosh.	
Called By		
Function	**Where Described**	
SaveGunneryTargets	See Section 2.13.8.12.2.	

Table 2.13-196: WriteGunneryTargets Information.

2.13.8.12.4 ReadGunneryTargets

ReadGunneryTargets reads in gunnery target information from *theFile*. The function call is ReadGunneryTargets(theFile). Table 2.13-197 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
gt	TargetHandle	Development:SIMNET:MCC:SCC:target.h
old_gt	TargetHandle	Development:SIMNET:MCC:SCC:target.h
f	GunneryTargetDescriptor	Development:SIMNET:MCC:SCC:target.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numTargets	int	Standard C type.
i	int	Standard C type.
newTargetNumber	short	Standard C type.
targetTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h

Calls	
Function	**Where Described**
FSRead	Standard File Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
NewTarget	See Section 2.13.6.2.1.
InstallScrollTableEntry	See Section 2.22.1.21.1.
UploadTarget	See Section 2.13.6.2.5.
Called By	
Function	**Where Described**
LoadGunneryTargets	See Section 2.13.8.12.1.

Table 2.13-197: ReadGunneryTargets Information.

2.13.8.12.5 WipeTargetTable

WipeTargetTable removes all the gunnery target table entries. The function call is WipeTargetTable(). Table 2.13-198 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register int	Standard C type.
targetByNumber	extern array of TargetHandle	Development:SIMNET:MCC:SCC:target.h
tgt	register pointer to TargetHandle	Development:SIMNET:MCC:SCC:target.h
Calls		
Function	**Where Described**	
RemoveTarget	See Section 2.13.6.2.2.	
Called By		
Function	**Where Described**	
LoadGunneryTargets	See Section 2.13.8.12.1.	

Table 2.13-198: WipeTargetTable Information.

2.13.8.12.6 LoadPresets

LoadPresets loads in combat vehicle descriptions contained in a prepared file. The function call is LoadPresets(). Table 2.13-199 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
where	Point	Development:THINK C: Mac #includes:MacTypes.h
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	FInfo	Development:THINK C: Mac #includes:FileMgr.h
theFile	int	Standard C type.
theTypeList	SFTypeList	Development:THINK C: Mac #includes:StdFilePkg.h
count	long	Standard C type.
Calls		
Function	**Where Described**	
SFGetFile	Standard Standard File Package function for Macintosh.	
SetCursor	Standard Quickdraw function for Macintosh.	
FSOpen	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
FSRead	Standard File Manager function for Macintosh.	
FlushVol	Standard File Manager function for Macintosh.	
Called By		
Function	**Where Described**	
ProcessCommandShiftOption Sequence	See Section 2.13.8.15.4.	

Table 2.13-199: LoadPresets Information.

2.13.8.12.7 SavePresets

SavePresets saves the vehicle descriptions in a file selected by the user. The function call is SavePresets(). Table 2.13-200 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
where	Point	Development:THINK C: Mac #includes:MacTypes.h
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	FInfo	Development:THINK C: Mac #includes:FileMgr.h
theFile	int	Standard C type.
count	long	Standard C type.

Calls	
Function	**Where Described**
SFPutFile	Standard Standard File Package function for Macintosh.
GetFInfo	Standard File Manager function for Macintosh.
SetCursor	Standard Quickdraw function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
Create	Standard File Manager function for Macintosh.
FSOpen	Standard File Manager function for Macintosh.
FSWrite	Standard File Manager function for Macintosh.
SetEOF	Standard File Manager function for Macintosh.
FSClose	Standard File Manager function for Macintosh.
FlushVol	Standard File Manager function for Macintosh.
Called By	
Function	**Where Described**
ProcessCommandShiftOption Sequence	See Section 2.13.8.15.4.

Table 2.13-200: SavePresets Information.

2.13.8.13 init.c

Development:SIMNET:MCC:SCC:init.c

init.c implements the initialization overview dialog, presenting a choice of elements to be initialized from the SCC. Table 2.13-201 describes the variables used by init.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
initSelection	char	Standard C type.
initTOCDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
initALOCDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cssTypeDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
initArtyDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
initCASDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
passwordDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
allocTableDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
placeSelectDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
cewAllocDialogOptionalElementInfo	array of OptionalElementInfo	Development:SIMNET:MCC: SCC:SCCMac.h
initOverviewFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
initOverviewFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
initOverviewDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-201: init.c Variable Information.

2.13.8.13.1 InitOverviewFetch

InitOverviewFetch reads the initialization overview controls into the dialog fields to display to the user. Controls are disabled if the section is not to be included, or if it is already initialized. The function call is InitOverviewFetch(dialog). Table 2.13-202 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
DisableControl	See Section 2.22.1.7.1.	

Table 2.13-202: InitOverviewFetch Information.

2.13.8.13.2 InitOverviewEvent

InitOverviewEvent is called when a button in the initialization overview dialog is pressed. Controls of the selected dialog are enabled (unless the Help or Goo buttons were pressed). The item number of the button pressed is returned. The function call is InitOverviewEvent(dialog, itemNo). Table 2.13-203 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of button pressed
Calls		
Function	**Where Described**	
EnableControl	See Section 2.22.1.7.2.	

Table 2.13-203: InitOverviewEvent Information.

2.13.8.13.3 InitOverviewBranch

InitOverviewBranch is used when there is a list of items to select. It checks the type of item selected and brings up the next dialog, depending on the item type. Returns a handle to the next dialog to pop up. The function call is InitOverviewBranch(dialog). Table 2.13-204 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Return Values		
Return Value	**Type**	**Meaning**
&allocTableDialog	pointer to DialogNodeDefn	handle of alloc table dialog
&placeSelectDialog	pointer to DialogNodeDefn	handle of the place select dialog
&passwordDialog	pointer to DialogNodeDefn	handle of the password dialog
&bmOverviewDialog	pointer to DialogNodeDefn	handle of the bm overview dialog
optionalElementInfo[initSelection].seq	pointer to DialogNodeDefn	handle of the optional element info dialogs

Table 2.13-204: InitOverviewBranch Information.

2.13.8.14 lock.c

Development:SIMNET:MCC:SCC:lock.c

lock.c implements a BattleMaster-controlled locking of the SCC Macintosh. When Cmd-Shift-Opt-L is pressed, a dialog is displayed indicating that the console is locked. All subsequent keystrokes and mouse clicks are ignored, except another Cmd-Shift-Opt-L, which causes the dialog to be removed. Table 2.13-205 describes the varibles used by lock.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
consoleLockDialog	DialogPtr	Development:THINK C: Mac #includes:DialogMgr.h

Table 2.13-205: lock.c Variable Information.

2.13.8.14.1 ToggleConsoleLock

ToggleConsoleLock locks and unlocks the console. The function call is ToggleConsoleLock(). Table 2.13-206 describes the functions called using this function.

Calls	
Function	**Where Described**
DisposDialog	Standard Dialog Manager function for Macintosh.
GetNewDialog	Standard Dialog Manager function for Macintosh.
SetWRefCon	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
ProcessCommandShiftOptionSequence	See Section 2.13.8.15.4.

Table 2.13-206: ToggleConsoleLock Information.

2.13.8.14.2 LockDialogEvent

LockDialogEvent is called when the lock dialog gets an event. The function call is LockDialogEvent(window, theEvent). Table 2.13-207 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
itemHit	int	Standard C type.
Calls		
Function	**Where Described**	
DialogSelect	Standard Dialog Manager function for Macintosh.	
SysBeep	Standard Operating System Utility function for Macintosh.	

Table 2.13-207: LockDialogEvent Information.

2.13.8.15 main.c

Development:SIMNET:MCC:SCC:main.c

main.c contains the SCC application's program entry point and main event loop.

2.13.8.15.1 main

main is the program entry point for the SCC application. The function call is main(). Table 2.13-208 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
startDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
Calls		
Function	**Where Described**	
SetUp	See Section 2.13.8.17.1.	
LoadCannedData	See Section 2.13.8.15.2.	
ProcessRequest	See Section 2.13.8.6.1.	
SetUpAppleTalk	See Section 2.22.1.3.1.	
DownloadTerrain	See Section 2.13.8.6.3.	
DownloadOptions	See Section 2.13.8.6.4.	
DownloadSimulators	See Section 2.22.2.3.1.	
MenuBarTitle	See Section 2.22.1.43.1.	
InstallClock	See Section 2.22.1.6.4.	
StartDialogSeq	See Section 2.22.1.39.1.	
MainEventLoop	See Section 2.13.8.15.3.	

Table 2.13-208: main Information.

2.13.8.15.2 LoadCannedData

LoadCannedData loads canned data into the data structures. This function is only compiled if VERSION is FULDAGAP. The function call is LoadCannedData(). Table 2.13-209 describes the functions called using this function.

Calls	
Function	**Where Described**
LoadCannedExercise	See Section 2.13.8.11.1.
LoadOptionsPermitted	See Section 2.13.8.11.7.
LoadCannedCSS	See Section 2.13.3.1.1.
LoadCannedTOC	See Section 2.13.1.2.1.
LoadCannedALOC	See Section 2.13.1.1.1.
LoadCannedCAS	See Section 2.13.4.1.1.
LoadCannedFSE	See Section 2.13.4.2.1.
LoadCannedSimulators	See Section 2.22.2.6.1.
Called By	
Function	**Where Described**
main	See Section 2.13.8.15.1.

Table 2.13-209: LoadCannedData Information.

2.13.8.15.3 MainEventLoop

MainEventLoop processes events off of the queue. The function call is MainEventLoop(). Table 2.13-210 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
passwordDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
myEvent	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
SystemTask	Standard Desk Manager function for Macintosh.	
UpdateClock	See Section 2.22.1.6.2.	
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.	
ProcessCommandShiftOption Sequence	See Section 2.13.8.15.4.	
WindowEvent	See Section 2.22.1.46.2.	
NetworkEventHandler	See Section 2.22.1.3.5.	
UpdateDisplacement	See Section 2.13.8.10.20.	
Called By		
Function	**Where Described**	
main	See Section 2.13.8.15.1.	

Table 2.13-210: MainEventLoop Information.

2.13.8.15.4 ProcessCommandShiftOptionSequence

ProcessCommandShiftOptionSequence determines what function to execute depending on which Cmd-Shift-Opt-<letter> combination has been pressed. *keypressed* is the letter that is pressed at the same time as the Command, Shift and Option keys. The function call is ProcessCommandShiftOptionSequence(keyPressed). Table 2.13-211 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
keyPressed	register int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dispUnitSelection	extern int	Standard C type.
Calls		
Function	**Where Described**	
SavePresets	See Section 2.13.8.12.7.	
SetCursor	Standard Quickdraw function for Macintosh.	
LoadPresets	See Section 2.13.8.12.6.	
SaveGunneryTargets	See Section 2.13.8.12.2.	
ToggleConsoleLock	See Section 2.13.8.14.1.	
ATPCloseSocket	Standard Appletalk Manager function for Macintosh.	
ExitToShell	Standard Segment Loader function for Macintosh.	
LoadGunneryTargets	See Section 2.13.8.12.1.	
ShowVersions	See Section 2.22.1.44.1.	
CompleteDisplacement	See Section 2.13.8.10.1.	
Called By		
Function	**Where Described**	
MainEventLoop	See Section 2.13.8.15.3.	

Table 2.13-211: ProcessCommandShiftOptionSequence Information.

2.13.8.16 master.c

Development:SIMNET:MCC:SCC:master.c

master.c implements the BattleMaster overview dialog, presenting a choice of activities for the BattleMaster. Table 2.13-212 describes the variables used by master.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
passwordBuffer	array of bmPasswordLength char	Standard C type.
passwordCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
passwordField	TextFieldDefn	Development:SIMNET:libmac: dialog.h
passwordFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
passwordDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
bmSelection	char	Standard C type.
stopDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
reconstDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
moreSortiesDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
displaceDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
bmFunctions	array of numberBMFunctions struct bmFunctions	
bmOverviewFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
bmOverviewFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
bmOverviewDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
stopFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
stopDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.13-212: master.c Variable Information.

2.13.8.16.1 PasswordFetch

PasswordFetch reads the password buffer into the Password dialog fields. The function call is PasswordFetch(dialog). Table 2.13-213 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h

Table 2.13-213: PasswordFetch Information.

2.13.8.16.2 PasswordEvent

PasswordEvent is called when a button in the password dialog is pressed. The function call is PasswordEvent(dialog, itemNo). Table 2.13-214 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cp1	register pointer to char	Standard C type.
cp2	register pointer to char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of button pressed
Calls		
Function	**Where Described**	
ShowCaution	See Section 2.22.1.4.1.	
SelIText	Standard Dialog Manager function for Macintosh.	

Table 2.13-214: PasswordEvent Information.

2.13.8.16.3 BMOverviewFetch

BMOverviewFetch reads the selections and sets up the controls for the battle master overview dialog fields to display to the user. The function call is BMOverviewFetch(dialog). Table 2.13-215 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
j	int	Standard C type.
supplyDepotsInited	extern char	Standard C type.
Calls		
Function	**Where Described**	
DisableControl	See Section 2.22.1.7.1.	

Table 2.13-215: BMOverviewFetch Information.

2.13.8.16.4 BMOverviewEvent

BMOverviewEvent is called when a button in the battle master overview dialog is pressed. The item number of the button pressed is returned. The function call is BMOverviewEvent(dialog, itemNo). Table 2.13-216 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	do nothing upon return
itemNo	int	resource id of button pressed
Calls		
Function	**Where Described**	
ShowTargetTable	See Section 2.13.6.4.2.	
GetDItem	Standard Dialog Manager function for Macintosh.	
SetCtlValue	Standard Control Manager function for Macintosh.	
DisableControl	See Section 2.22.1.7.1.	
DialogSeqNext	See Section 2.22.1.39.4.	
HiliteControl	Standard Control Manager function for Macintosh.	

Table 2.13-216: BMOverviewEvent Information.

2.13.8.16.5 StopEvent

StopEvent is called when a button in the stop dialog is pressed. It returns the item number of the button pressed. The function call is StopEvent(dialog, itemNo). Table 2.13-217 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of button pressed

Calls	
Function	**Where Described**
ATPPut	See Section 2.22.1.3.3.
ATPCloseSocket	Standard Appletalk Manager function for Macintosh.
Restart	Standard Operating System Utility function for Macintosh.

Table 2.13-217: StopEvent Information.

2.13.8.17 setup.c

Development:SIMNET:MCC:SCC:setup.c

setup.c initializes the SCC application at start-up.

2.13.8.17.1 SetUp

SetUp initializes the SIMNET Control Console application. The toolbox is initialized, the SIMNET Resources and SCC Picture Files are opened, the Help manager is initialized, and storage is allocated for the dialog states. The function call is SetUp(). Table 2.13-218 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
errCode	int	Standard C type.
port	GrafPort	Development:THINK C: Mac #includes:Quickdraw.h
finalTicks	long	Standard C type.

Calls	
Function	**Where Described**
InitToolbox	See Section 2.22.1.20.1.
ZoomInit	See Section 2.22.1.47.1.
GetCursor	Standard Quickdraw function for Macintosh.
OpenResFile	Standard Resource Manager function for Macintosh.
MenuBarTitle	See Section 2.22.1.43.1.
Delay	Standard Operating System Utility function for Macintosh.
ExitToShell	Standard Segment Loader function for Macintosh.
DeepShit	See Section 2.22.1.8.2.
helpinit	See Section 2.22.1.19.1.
SystemFailure	See Section 2.22.1.8.1.
NewPtr	Standard Memory Manager function for Macintosh.
OpenPort	Standard Quickdraw function for Macintosh.
SetUpAlloc	See Section 2.13.2.1.1.
SetUpDisplacement	See Section 2.13.8.10.13.
SetUpOptionalElements	See Section 2.13.8.11.8.
SetUpPlace	See Section 2.13.2.2.1.
SetUpTargets	See Section 2.13.6.4.1.
SetUpTrucks	See Section 2.13.3.4.1.
ClosePort	Standard Quickdraw function for Macintosh.
SetUpSimulators	See Section 2.22.2.9.1.

Called By	
Function	**Where Described**
main	See Section 2.13.8.15.1.

Table 2.13-218: SetUp Information.

2.13.8.18 string.c

Development:SIMNET:MCC:SCC:string.c

string.c contains a routine for copying Pascal-format strings.

2.13.8.18.1 Pstrcpy

Pstrcpy copies a Pascal string *src* to another Pascal string*dest* and returns a pointer to the result. The function call is Pstrcpy(dest, src). Table 2.13-219 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dest	register pointer to char	Standard C type.
src	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
len	register int	Standard C type.
i	register int	Standard C type.
result	register pointer to char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
result	pointer to char	pointer to Pascal string copies into *dest*.

Table 2.13-219: Pstrcpy Information.

2.14 The Place (Placement) Console

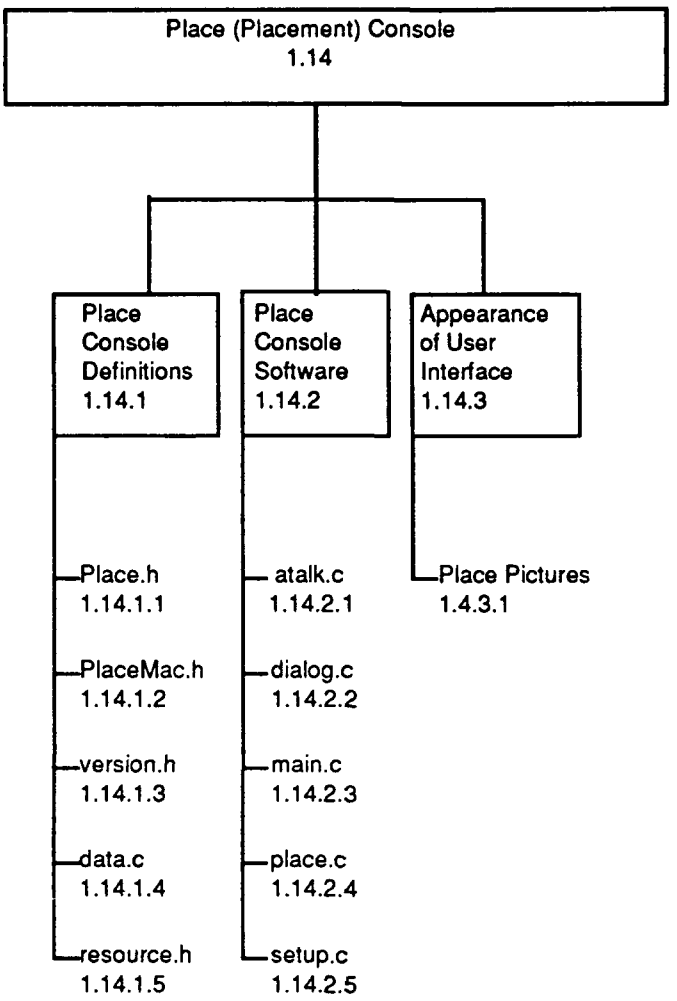


Figure 2.14-1: Place (Placement) Console Structure.

2.14.1 Place Console Definitions

2.14.1.1 Place.h

Development:SIMNET:MCC:Place:Place.h

Place.h defines the representation of information communicated between the Place Macintosh application and the MCC host.

2.14.1.2 PlaceMac.h
Development:SIMNET:MCC:Place:PlaceMac.h

PlaceMac.h defines types, constants and routines used within the Place Macintosh application. Table 2.14-1 describes the variables used by PlaceMac.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
options	OptionalElements	Development:SIMNET:MCC: include:options.h
seqDialog	extern pointer to DialogSeqState	Development:SIMNET:libmac: sequence.h
watchCursorHandle	extern CursHandle	Development:THINK C: Mac #includes:Quickdraw.h

Table 2.14-1: PlaceMac.h Variable Information.

2.14.1.3 version.h
Development:SIMNET:MCC:Place:version.h

version.h defines constants that determine which version of the Place application is compiled.

2.14.1.4 data.c
Development:SIMNET:MCC:Place:data.c

data.c defines various data structures used by the Place application. Table 2.14-2 describes the variables used by data.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
application	pointer to char	Standard C type.
authors	pointer to char	Standard C type.
copyright	pointer to char	Standard C type.
options	OptionalElements	Development:SIMNET:MCC: include:options.h
seqDialog	pointer to DialogSeqState	Development:SIMNET:libmac: sequence.h
watchCursorHandle	CursHandle	Development:THINK C: Mac #includes:Quickdraw.h

Table 2.14-2: data.c Variable Information.

2.14.1.5 resource.h
Development:SIMNET:MCC:Place:resource.h

resource.h defines the numbers of the resources present in the Place Pictures resource file.

2.14.2 Place Console Software

2.14.2.1 atalk.c
Development:SIMNET:MCC:Place:atalk.c

atalk.c contains routines supporting AppleTalk communication between the Place Macintosh application and the MCC host process.

2.14.2.1.1 ProcessRequest

ProcessRequest processes a request received from the MCC host. The function call is ProcessRequest(hdl). Table 2.14-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
hdl	Handle	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	register Ptr	Development:THINK C: Mac #includes:MacTypes.h
errCode	int	Standard C type.
Calls		
Function	**Where Described**	
SetDateTime	Standard Operating System Utility function for the Macintosh.	
ATPResponse	Standard Appletalk Manager function for the Macintosh.	
Restart	Standard Operating System Utility function for the Macintosh.	
SimulatorAlloced	See Section 2.14.2.4.8.	
ProcessSimPlacedRequest	See Section 2.22.2.3.4.	
SimulatorPlaced	See Section 2.14.2.4.9.	
ATPError	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
main	See Section 2.14.2.3.1.	

Table 2.14-3: ProcessRequest Information.

2.14.2.1.2 DownloadParameters

DownloadParameters downloads information about grid zones, participating units, and simultors from the host at the start of an exercise. The function call is DownloadParameters(). Table 2.14-4 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
rsp	PlaceStartResponse	Development:SIMNET:MCC:Place:Place.h
Calls		
Function	**Where Described**	
DownloadTerrainMap	See Section 2.22.1.3.4.	
ATPPut	See Section 2.22.1.3.3.	
DownloadSimulators	See Section 2.22.2.3.1.	
Called By		
Function	**Where Described**	
main	See Section 2.14.2.3.1.	

Table 2.14-4: DownloadParameters Information.

2.14.2.2 dialog.c

Development:SIMNET:MCC:Place:dialog.c

dialog.c contains miscellaneous routines supporting dialog implementations. Table 2.14-5 describes the variables used by dialog.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
tableRowRext	Rect	Development:THINK C:Mac #includes:MacTypes.h
popUpWindow	WindowPtr	Development:THINK C:Mac #includes:WindowMgr.h

Table 2.14-5: dialog.c Variable Information.

2.14.2.2.1 ZoomUpFromTableEntry

ZoomUpFromTableEntry zooms from a row in a table, given by *box*, to a pop up window. The function call is ZoomUpFromTableEntry(box). Table 2.14-6 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
box	register pointer to Rect	Development:THINK C:Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for the Macintosh.	

Called By	
Function	**Where Described**
PlaceTableSelect	See Section 2.14.2.4.6.

Table 2.14-6: ZoomUpFromTableEntry Information.

2.14.2.2.2 ZoomDownToTableEntry

ZoomDownToTableEntry zooms from a pop up window to a row within a table. The function call is ZoomDownToTableEntry(). Table 2.14-7 describes the functions called by this function.

Calls	
Function	**Where Described**
HideWindow	Standard Window Manager function for the Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
Called By	
Function	**Where Described**
PlaceComplete	See Section 2.14.2.4.5.

Table 2.14-7: ZoomDownToTableEntry Information.

2.14.2.3 main.c

Development:SIMNET:MCC:Place:main.c

main.c contains the Place application's program entry point and main event loop.

2.14.2.3.1 main

main is the program entry point for the Place console application. The function call is main(). Table 2.14-8 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
placeSelectDialog	extern DialogNodeDefn	Development:SIMNET:libmac:sequence.h
Calls		
Function	**Where Described**	
SetUp	See Section 2.14.2.5.1.	
LoadCannedTerrainMap	See Section 2.14.2.3.2.	
LoadCannedSimulators	See Section 2.22.2.6.1.	
ProcessRequest	See Section 2.14.2.1.1.	
SetUpAppleTalk	See Section 2.22.1.3.1.	
DownloadParameters	See Section 2.14.2.1.2	
MenuBarTitle	See Section 2.22.1.43.1	
InstallClock	See Section 2.22.1.6.4.	
StartDialogSeq	See Section 2.22.1.39.1.	
MainEventLoop	See Section 2.14.2.3.3.	

Table 2.14-8: main Information.

2.14.2.3.2 LoadCannedTerrainMap

LoadCannedTerrainMap loads the terrain map structure with dummy data for the demo version of the program. The function is only compiled if VERSION is FULDAGAP. The function call is LoadCannedTerrainMap(). Table 2.14-9 describes the functions calling this function.

Called By	
Function	**Where Described**
main	See Section 2.14.2.3.1.

Table 2.14-9: LoadCannedTerrainMap Information.

2.14.2.3.3 MainEventLoop

MainEventLoop processes events off the queue. The function call is MainEventLoop(). Table 2.14-10 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
myEvent	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
SystemTask	Standard Desk Manager function for the Macintosh.	
UpdateClock	See Section 2.22.1.6.2.	
GetNextEvent	Standard Toolbox Event Manager function for the Macintosh.	
NetworkEventHandler	See Section 2.22.1.3.5.	
ATPCloseSocket	Standard Appletalk Manager function for the Macintosh.	
ExitToShell	Standard Segment Loader function for the Macintosh.	
ShowVersions	See Section 2.22.1.44.1.	
WindowEvent	See Section 2.22.1.46.2.	
Called By		
Function	**Where Described**	
main	See Section 2.14.2.3.1.	

Table 2.14-10: MainEventLoop Information.

2.14.2.4 place.c

Development:SIMNET:MCC:Place:place.c

place.c implements the user interface for placing vehicle simulators. Table 2.14-11 describes the variables used by place.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
placeCompany	char	Standard C type.
placeTableDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
placeTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
placeColumns	extern array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
placeSelectFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
placeSelectFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
placeSelectDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h
placeColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
placeTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
placeTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
placeTableDialog	DialogNodeDefn	Development:SIMNET:libmac: sequence.h

Table 2.14-11: place.c Variable Information.

2.14.2.4.1 SetUpPlace

SetUpPlace sets up the dialog for placing vehicles. The function call is SetUpPlace(). Table 2.14-12 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for the Macintosh.
TextSize	Standard Quickdraw function for the Macintosh.
StringWidth	Standard Quickdraw function for the Macintosh.
Called By	
Function	**Where Described**
SetUp	See Section 2.14.2.5.1.

Table 2.14-12: SetUpPlace Information.

2.14.2.4.2 PlaceSelectFetch

PlaceSelectFetch reads the participating organizational units into the Place Select dialog to display to the user. The Place Select dialog allows the user to select the company whose simulators are to be placed. The routine also disables radio buttons for optional elements not included in the exercise and for organizational units which are not participating. The function call is PlaceSelectFetch(dialog). Table 2.14-13 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Calls		
Function	**Where Described**	
DisableControl	See Section 2.22.1.7.1.	

Table 2.14-13: PlaceSelectFetch Information.

2.14.2.4.3 PlaceSelectEvent

PlaceSelectEvent is called when a button in the Place Select dialog is pressed. When a valid button is pressed, the routine enables control of the Go button. The item number of the button pressed is returned. The function call is PlaceSelectEvent(dialog, itemNo). Table 2.14-14 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
EnableControl	See Section 2.22.1.7.2.	

Table 2.14-14: PlaceSelectEvent Information.

2.14.2.4.4 PlaceTableFetch

PlaceTableFetch reads the selected company's simulators into the Place Table dialog. The Place Table dialog displays the table of simulators for the selected company. The function call is PlaceTableFetch(dialog). Table 2.14-15 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h

Calls	
Function	**Where Described**
SetText	See Section 2.22.1.11.8.
CompanyName	See Section 2.22.2.9.4.
SimTableSetup	See Section 2.22.2.10.1.

Table 2.14-15: PlaceTableFetch Information.

2.14.2.4.5 PlaceComplete

PlaceComplete is called when a valid Place Table dialog selection has been made and the dialog should be taken down. The function call is PlaceComplete(sim, success). Table 2.14-16 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sim	pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
success	int	Standard C type.

Calls	
Function	**Where Described**
UpdateScrollTableEntry	See Section 2.22.1.31.1.
ZoomDownToTableEntry	See Section 2.14.2.2.2.

Table 2.14-16: PlaceComplete Information.

2.14.2.4.6 PlaceTableSelect

PlaceTableSelect is called when an entry in the Place Table has been selected. The routine checks the validity of the selection and pops up the dialog for placing the vehicle. The function call is PlaceTableSelect(defn, row, box, event). Table 2.14-17 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h

Calls	
Function	**Where Described**
ScrollTableRowToEntry	See Section 2.22.1.34.1.
SetScrollTableSelection	See Section 2.22.1.37.2.
SimTableEntry	See Section 2.22.2.10.3.
ShowCaution	See Section 2.22.1.4.1.
ShowVehicleDialog	See Section 2.22.2.9.5.
ZoomUpFromTableEntry	See Section 2.14.2.2.1.

Table 2.14-17: PlaceTableSelect Information.

2.14.2.4.7 PlaceTableEvent

PlaceTableEvent is called when an entry in the Place Table dialog is selected. If the Overview button was pressed, the scroll table is discarded and the main screen is displayed. The item number of the button is returned. The function call is PlaceTableEvent(dialog, itemNo). Table 2.14-18 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	resource id of the button hit
Calls		
Function	**Where Described**	
DisposeScrollTable	See Section 2.22.1.30.2.	

Table 2.14-18: PlaceTableEvent Information.

2.14.2.4.8 SimulatorAlloced

SimulatorAlloced notes that a simulator has been allocated by the SCC console. If a table of simulators is being displayed, the screen is updated. The function call is SimulatorAlloced(vehicle, company). Table 2.14-19 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
vehicle	int	Standard C type.
company	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h

Calls	
Function	**Where Described**
SimTableUpdate	See Section 2.22.2.10.2.
Called By	
Function	**Where Described**
ProcessRequest	See Section 2.14.2.1.1.

Table 2.14-19: SimulatorAlloced Information.

2.14.2.4.9 SimulatorPlaced

SimulatorPlaced notes that a simulator has been placed by another console. If a table of simulators is being displayed, the screen is updated. The function call is SimulatorPlaced(sim). Table 2.14-20 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
sim	register pointer to SimDescriptor	Development:SIMNET:MCC: libsim:libsim.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
placeTableDialog	extern DialogNodeDefn	Development:SIMNET:libmac: sequence.h
placeTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
Calls		
Function	**Where Described**	
SimTableUpdate	See Section 2.22.2.10.2.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.14.2.1.1.	

Table 2.14-20: SimulatorPlaced Information.

2.14.2.5 setup.c

Development:SIMNET:MCC:Place:setup.c

setup.c initializes the Place application at start-up.

2.14.2.5.1 SetUp

SetUp initializes the Place console application. The toolbox is initialized, the SIMNET Resources and Place Pictures files are opened, the Help Manager is initialized, and storage is allocated for dialog states. The function call is SetUp(). Table 2.14-21 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
errCode	int	Standard C type.
port	GrafPort	Development:THINK C: Mac #includes:Quickdraw.h
finalTicks	long	Standard C type.
Calls		
Function	**Where Described**	
InitToolbox	See Section 2.22.1.20.1.	
ZoomInit	See Section 2.22.1.47.1.	
GetCursor	Standard Toolbox Utility function for the Macintosh.	
OpenResFile	Standard Resource Manager function for the Macintosh.	
MenuBarTitle	See Section 2.22.1.43.1.	
Delay	Standard Operating System Utility function for the Macintosh.	
ExitToShell	Standard Segment Loader function for the Macintosh.	
DeepShit	See Section 2.22.1.8.?.	
helpinit	See Section 2.22.1.19.1.	
SystemFailure	See Section 2.22.1.8.1.	
NewPtr	Standard Memory Manager function for the Macintosh.	
OpenPort	Standard Quickdraw function for the Macintosh.	
SetUpPlace	See Section 2.14.2.4.1.	
ClosePort	Standard Quickdraw function for the Macintosh.	
SetUpSimulators	See Section 2.22.2.9.1.	
Called By		
Function	**Where Described**	
main	See Section 2.14.2.3.1.	

Table 2.14-21: SetUp Information.

2.14.3 Appearance of User Interface

2.14.3.1 Place Pictures

Development:SIMNET:MCC:Place:Place Pictures

Place Pictures contains resources that determine the appearance of the Place application's user interface.

2.15 The Admin Console

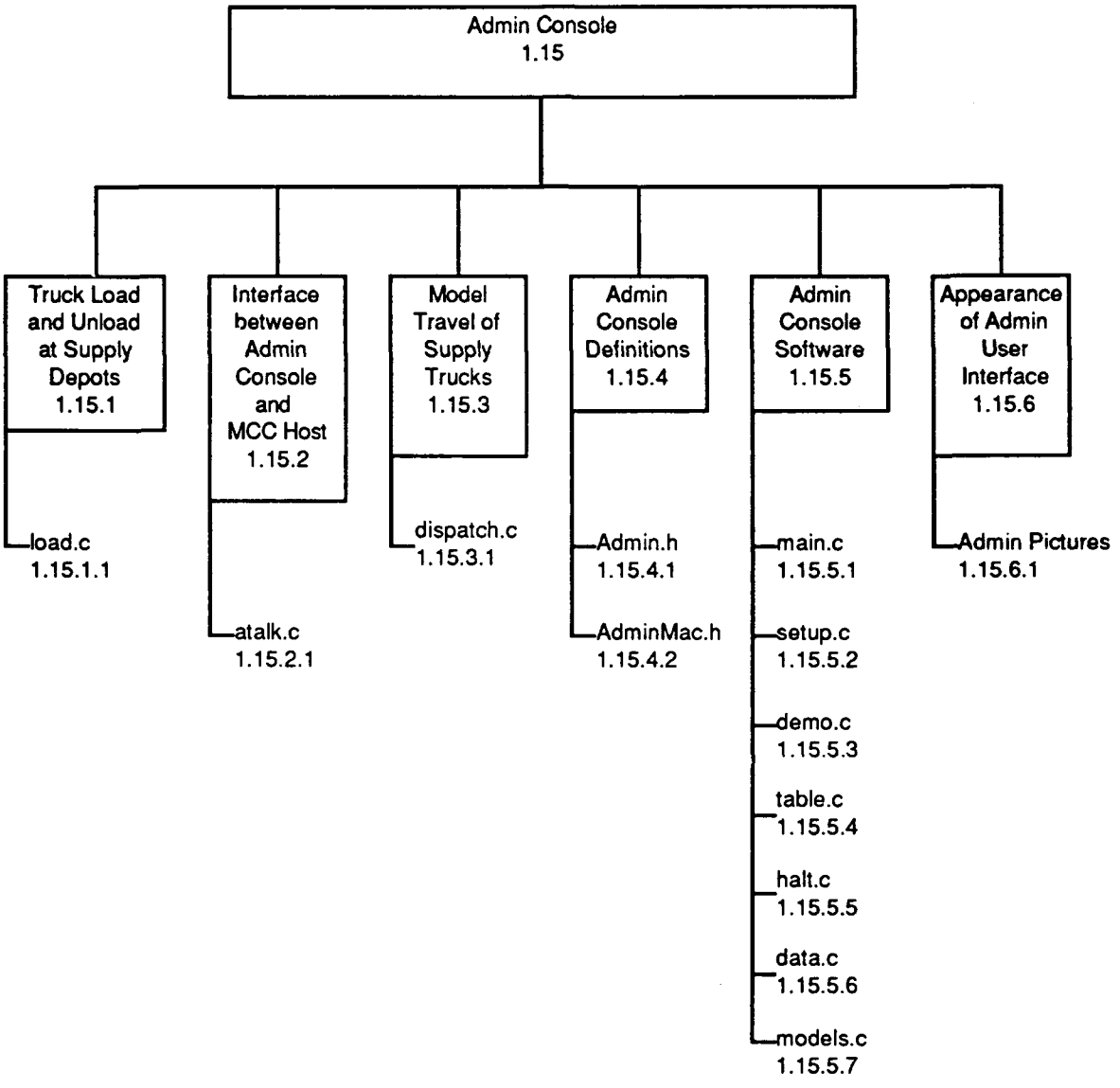


Figure 2.15-1: Admin Console Structure.

2.15.1 Truck Load and Unload at Supply Depots

2.15.1.1 load.c
Development:SIMNET:MCC:Admin:load.c

load.c implements the user interface for loading supply trucks with ammunition and fuel. Table 2.15-1 describes the variables used by load.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truckParty	TransferParty	Development:SIMNET:MCC: libsupply:libsupply.h
depotParty	TransferParty	Development:SIMNET:MCC: libsupply:libsupply.h
fuelQuantityBuffer	int	Standard C type.
fuelLoadCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
fuelLoadQuantityField	NumberFieldDefn	Development:SIMNET:libmac: dialog.h
fuelTruckLoadFields	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
fuelTruckLoadDialog	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.15-1: load.c Variable Information.

2.15.1.1.1 ShowLoadDialog

ShowLoadDialog pops up the Load dialog. The Load dialog is used for loading a truck at a supply depot. The function call is ShowLoadDialog(truck, window, rect). Table 2.15-2 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
rect	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ShowTransferAmmoDialog	See Section 2.22.3.4.1.	
ShowDialog	See Section 2.22.1.11.1.	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
TruckTableEvent	See Section 2.15.5.4.4.	

Table 2.15-2: ShowLoadDialog Information.

2.15.1.1.2 AmmoTruckLoadComplete

AmmoTruckLoadComplete is called when the OK button is hit on the Ammo Transfer dialog. This routine takes down the Ammo Transfer dialog, zooms to the Truck Table dialog, and updates the loaded truck's row in the Truck Table. The function call is AmmoTruckLoadComplete(). Table 2.15-3 describes the functions called using this function.

Calls	
Function	**Where Described**
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
UpdateTruckTableRow	See Section 2.15.5.4.14.

Table 2.15-3: AmmoTruckLoadComplete Information.

2.15.1.1.3 FuelTruckLoadFetch

FuelTruckLoadFetch reads the current values of the fuelQuantityBuffer data structure into the Fuel Truck Load dialog fields to display to the user. The function call is FuelTruckLoadFetch(dialog). Table 2.15-4 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
string	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
SellText	Standard Dialog Manager function for Macintosh.	
SetText	See Section 2.22.1.11.8.	

Table 2.15-4: FuelTruckLoadFetch Information.

2.15.1.1.4 FuelTruckLoadEvent

FuelTruckLoadEvent is called when a button in the Fuel Truck Load dialog is pressed. When the OK button is pressed, the routine checks the validity of the data entered, updates the fuelQuantityBuffer of the current truck. When the Cancel button is pressed, the Fuel Truck Load dialog is closed. The function call is FuelTruckLoadEvent(dialog, itemNo). Table 2.15-5 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
0	int	Invalid data entered.

Calls	
Function	**Where Described**
CheckMandatoryFields	See Section 2.22.1.13.1.
CloseLoadDialog	See Section 2.15.1.1.6.

Table 2.15-5: FuelTruckLoadEvent Information.

2.15.1.1.5 UpdateLoadDialog

UpdateLoadDialog updates the load displayed in the Truck Load dialog. The function call is UpdateLoadDialog(). Table 2.15-6 describes the functions called using this function.

Calls	
Function	**Where Described**
UpdateAmmoTransferLoads	See Section 2.22.3.4.4.

Table 2.15-6: UpdateLoadDialog Information.

2.15.1.1.6 CloseLoadDialog

CloseLoadDialog is called to close the Truck Load dialog. The routine zooms back down to the Truck Table dialog and updates the dispatched truck's row in the truck table dialog. The function call is CloseLoadDialog(). Table 2.15-7 describes the functions called using this function.

Calls	
Function	**Where Described**
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
ThrowDialog	See Section 2.22.1.11.3.
UpdateTruckTableRow	See Section 2.15.5.4.14.

Called By	
Function	**Where Described**
FuelTruckLoadEvent	See Section 2.15.1.1.4.

Table 2.15-7: CloseLoadDialog Information.

2.15.2 Interface between Admin Console and MCC Host

2.15.2.1 atalk.c

Development:SIMNET:MCC:Admin:atalk.c

atalk.c processes requests and responses received from the MCC host via the AppleTalk network. The functions in this file are only compiled if VERSION is APPLETALK.

2.15.2.1.1 ProcessRequest

ProcessRequest processes a request received from the MCC host. The function call is ProcessRequest(hdl). Table 2.15-8 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
hdl	Handle	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	register Ptr	Development:THINK C: Mac #includes:MacTypes.h
errCode	int	Standard C type.
i	int	Standard C type.
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
Calls		
Function	**Where Described**	
SetDateTime	Standard Operating System Utility function for Macintosh.	
DiscardPopUpDialog	See Section 2.15.5.7.9.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
InitTruckState	See Section 2.15.5.7.1.	
UpdateTruckTableRow	See Section 2.15.5.4.14.	
UpdatePopUpLoad	See Section 2.15.5.7.8.	
NotifyStatusChange	See Section 2.15.5.7.7.	
Restart	Standard Operating System Utility function for Macintosh.	
ATPResponse	Standard Appletalk Manager function for Macintosh.	
Called By		
Function	**Where Described**	
main	See Section 2.15.5.1.1.	

Table 2.15-8: ProcessRequest Information.

2.15.2.1.2 ProcessResponse

ProcessResponse processes a response received from the MCC host. The function call is ProcessResponse(hdl). Table 2.15-9 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
hdl	Handle	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	register Ptr	Development:THINK C: Mac #includes:MacTypes.h
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
Calls		
Function	**Where Described**	
PointToMapCoordinates	See Section 2.22.1.26.2.	
DiscardPopUpDialog	See Section 2.15.5.7.9.	
UpdateTruckTableRow	See Section 2.15.5.4.14.	

Table 2.15-9: ProcessResponse Information.

2.15.2.1.3 ReportDispatch

ReportDispatch reports to the host that a truck, *truck*, has been dispatched. The function call is ReportDispatch(truck). Table 2.15-10 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	AdminLogDispatchRequest	Development:SIMNET:MCC: Admin:Admin.h
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
TruckDispatchEvent	See Section 2.15.3.1.3.	

Table 2.15-10: ReportDispatch Information.

2.15.2.1.4 ReportHalt

ReportHalt reports to the host that a truck, *truck*, has halted. The function call is ReportHalt(truck). Table 2.15-11 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ab	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
req	pointer to AdminLogArriveRequest	Development:SIMNET:MCC: Admin:Admin.h
rsp	pointer to AdminLogArriveResponse	Development:SIMNET:MCC: Admin:Admin.h
i	register int	Standard C type.
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
NewPtr	Standard Memory Manager function for Macintosh.	
SetUpATPRequest	See Section 2.22.1.3.2.	
ATPRequest	Standard Appletalk Manager function for Macintosh.	
Called By		
Function	**Where Described**	
TruckHaltEvent	See Section 2.15.5.5.3.	
Update TruckLocation	See Section 2.15.5.7.3.	
DisableTruck	See Section 2.15.5.7.4.	

Table 2.15-11: ReportHalt Information.

2.15.3 Model Travel of Supply Trucks

2.15.3.1 dispatch.c

Development:SIMNET:MCC:Admin:dispatch.c

dispatch.c implements dialogs for dispatching and halting supply trucks. Table 2.15-12 describes the variables used by dispatch.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
assignmentBuffer	char	Standard C type.
supplyPtBuffer	char	Standard C type.
locationBuffer	MapCoordinates	Development:SIMNET:libmac: map.h
dispatchAmmoTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
assignmentFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
supplyPtFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
dispatchCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
locationField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
ammoTruckDispatchFields	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
fuelTruckDispatchFields	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
ammoTruckDispatchDialog	DialogDefn	Development:SIMNET:libmac: dialog.h
fuelTruckDispatchDialog	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.15-12: dispatch.c Variable Information.

2.15.3.1.1 ShowDispatchDialog

ShowDispatchDialog pops up the Dispatch dialog. The Dispatch dialog is used to order a truck to dispatch. The function call is ShowDispatchDialog(truck, window, rect). Table 2.15-13 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
rect	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
TruckTableEvent	See Section 2.15.5.4.5.	

Table 2.15-13: ShowDispatchDialog Information.

2.15.3.1.2 TruckDispatchFetch

TruckDispatchFetch reads the current values of the assignmentBuffer, supplyPtBuffer, locationBuffer, and text fields into the Truck Dispatch dialog fields to display to the user. The routine displays the load onboard the current truck in the dialog. The routine also disables radio buttons for companies not belonging to the same force as the currently chosen company. The function call is TruckDispatchFetch(dialog). Table 2.15-14 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
str	array of 100 char	Standard C type.
forceID	char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
UpdateDispatchLoad	See Section 2.15.3.1.5.	
DisableControl	See Section 2.22.1.7.1.	

Table 2.15-14: *TruckDispatchFetch* Information.

2.15.3.1.3 TruckDispatchEvent

TruckDispatchEvent is called when a button in the Truck Dispatch dialog is pressed. If the Dispatch button is pressed, the location and estimated arrival time are computed, any changes in the company assignment radio buttons are noted, and the truck is reported as having been dispatched. If the Cancel button is pressed, the dialog is closed. If the Compute ETA button is pressed, the arrival time is computed and displayed. The function call is TruckDispatchEvent(dialog, itemNo). Table 2.15-15 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 20 char	Standard C type.

Calls	
Function	**Where Described**
ComputeETA	See Section 2.15.3.1.4.
ReportDispatch	See Section 2.15.2.1.3.
CloseDispatchDialog	See Section 2.15.3.1.6.
DTGToString	See Section 2.22.1.15.1.
SetText	See Section 2.22.1.11.8.
SellText	Standard Dialog Manager function for Macintosh.
SetRadioButton	See Section 2.22.1.11.6.

Table 2.15-15: TruckDispatchEvent Information.

2.15.3.1.4 ComputeETA

ComputeETA computes the currently displayed truck's arrival time. The routine ensures that if the truck is not being returned to the supply point, a destination location is entered. The function call is ComputeETA(). Table 2.15-16 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
distance	long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Unable to compute ETA.
1	int	Successful.
Calls		
Function	**Where Described**	
ShowCaution	See Section 2.22.1.4.1.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
DistBetweeen2Pts	See Section 2.22.1.23.2.	
GetDateTime	Standard OperatingSystem Utility function for Macintosh.	
DTGElapsed	See Section 2.22.1.15.3.	
Called By		
Function	**Where Described**	
TruckDispatchEvent	See Section 2.15.3.1.3.	

Table 2.15-16: ComputeETA Information.

2.15.3.1.5 UpdateDispatchLoad

UpdateDispatchLoad updates the load displayed in the Truck Dispatch dialog. The function call is UpdateDispatchLoad(). Table 2.15-17 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str[50]	char	Standard C type.

Calls	
Function	**Where Described**
EmptyScrollTable	See Section 2.22.1.28.2.
FillAmmunitionList	See Section 2.22.1.5.1.
SetText	See Section 2.22.1.11.8.
Called By	
Function	**Where Described**
TruckDispatchFetch	See Section 2.15.3.1.2.

Table 2.15-17: UpdateDispatchLoad Information.

2.15.3.1.6 CloseDispatchDialog

CloseDispatchDialog is called to close a Truck Dispatch dialog. The function call is CloseDispatchDialog(). Table 2.15-18 describes the functions called using this function.

Calls	
Function	**Where Described**
DisposeScrollTable	See Section 2.22.1.30.2.
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
ThrowDialog	See Section 2.22.1.11.3.
UpdateTruckTableRow	See Section 2.15.5.4.14.
Called By	
Function	**Where Described**
TruckDispatchEvent	See Section 2.15.3.1.3.

Table 2.15-18: CloseDispatchDialog Information.

2.15.4 Admin Console Definitions

2.15.4.1 Admin.h
Development:SIMNET:MCC:Admin:Admin.h

Admin.h defines the representation of information communicated between the Admin Macintosh application and the MCC host.

2.15.4.2 AdminMac.h
Development:SIMNET:MCC:Admin:AdminMac.h

AdminMac.h defines types, constants and routines used within the Admin Macintosh application. Table 2.15-19 describes the variables used by AdminMac.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
ammoTrucks	extern array of TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
fuelTrucks	extern array of TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
ammoSupplyPoint	extern MapCoordinates	Development:SIMNET:libmac:map.h
ammoSupplyContent	extern Manifest	Development:SIMN ET:MCC:include:resupply.h
fuelSupplyPoint	extern MapCoordinates	Development:SIMNET:libmac:map.h
currentTruck	extern pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
zoomBaseWindow	extern WindowPtr	Development:THINK C:Mac #includes:WindowMgr.h
zoomBaseRect	extern Rect	Development:THINK C:Mac #includes:MacTypes.h
popUpDialogState	pointer to DialogState	Development:SIMNET:libmac:dialog.h
(updatePopUpDialog)()	extern void function call	Standard
(closePopUpDialog)()	extern void function call	Standard
battleScheme	extern char	Standard C type.
bnForceID	extern char	Standard C type.
companyForceID	extern array of maxNumberCompanies char	Standard C type.
startUpComplete	extern char	Standard C type.

Table 2.15-19: AdminMac.h VariableInformation.

2.15.5 Admin Console Software

2.15.5.1 main.c

Development:SIMNET:MCC:Admin:main.c

main.c contains the Admin application's program entry point and main event loop.

2.15.5.1.1 main

main is the application entry point. The function call is main(). Table 2.15-20 describes the functions called using this function.

Calls	
Function	**Where Described**
SetUp	See Section 2.15.5.2.1.
ProcessRequest	See Section 2.15.2.1.1.
SetUpAppleTalk	See Section 2.22.1.3.1.
DownloadTerrainMap	See Section 2.22.1.3.4.
DownloadInitialData	See Section 2.15.5.1.2.
LoadCannedData	See Section 2.15.5.1.1.
MenuBarTitle	See Section 2.22.1.43.1.
InstallClock	See Section 2.22.1.6.4.
ShowTruckTables	See Section 2.15.5.4.2.
MainEventLoop	See Section 2.15.5.1.3.

Table 2.15-20: main Information.

2.15.5.1.2 DownloadInitialData

DownloadInitialData downloads information about the depots and trucks. It is only compiled if VERSION is APPLETALK. The function call is DownloadInitialData(). Table 2.15-21 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
myEvent	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.	
NetworkEventHandler	See Section 2.22.1.3.5.	
Called By		
Function	**Where Described**	
main	See Section 2.15.5.1.1.	

Table 2.15-21: DownloadInitialData Information.

2.15.5.1.3 MainEventLoop

MainEventLoop processes incoming events from the queue. The function call is MainEventLoop(). Table 2.15-22 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
event	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
str	array of char	Standard C type.

Calls	
Function	**Where Described**
UpdateClock	See Section 2.22.1.6.2.
CheckTimers	See Section 2.15.5.7.2.
SystemTask	Standard Desk Manager function for Macintosh.
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
ATPCloseSocket	Standard Appletalk Manager function for Macintosh.
ExitToShell	Standard Segment Loader function for Macintosh.
ShowVersions	See Section 2.22.1.44.1.
GetDateTime	Standard Operating System Utility function for Macintosh.
WindowEvent	See Section 2.22.1.46.2.
NetworkEventHandler	See Section 2.22.1.3.5.
Called By	
Function	**Where Described**
main	See Section 2.15.5.1.1.

Table 2.15-22: MainEventLoop Information.

2.15.5.2 setup.c

Development:SIMNET:MCC:Admin:setup.c

setup.c initializes the Admin application at start-up.

2.15.5.2.1 SetUp

SetUp initializes the Admin/Log application. The Macintosh application environmnet is initialized, the SIMNET Resources and Admin Pictures files are opened, the random number generator is initialized, the tables of truck information are initialized, and the dialogs are initialized. The function call is SetUp(). Table 2.15-23 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
port	GrafPort	Development:THINK C: Mac #includes:Quickdraw.h
finalTicks	long	Standard C type.
i	register short	Standard C type.
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h

Calls	
Function	**Where Described**
InitToolbox	See Section 2.22.1.20.1.
ZoomInit	See Section 2.22.1.47.1.
OpenResFile	Standard Resource Manager function for Macintosh.
MenuBarTitle	See Section 2.22.1.43.1.
Delay	Standard Operating System Utility function for Macintosh.
ExitToShell	Standard Segment Loader function for Macintosh.
DeepShit	See Section 2.22.1.8.2.
GetDateTime	Standard Operating System Utility function for Macintosh.
InitTruckState	See Section 2.15.5.7.1.
NewPtr	Standard Memory Manager function for Macintosh.
OpenPort	Standard Quickdraw function for Macintosh.
SetUpTruckTables	See Section 2.15.5.4.1.
ClosePort	Standard Quickdraw function for Macintosh.
Called By	
Function	**Where Described**
main	See Section 2.15.5.1.1.

Table 2.15-23: SetUp Information.

2.15.5.3 demo.c

Development:SIMNET:MCC:Admin:demo.c

demo.c loads canned data into application data structures, in the standalone demo version of the Admin application.

2.15.5.3.1 LoadCannedData

LoadCannedData loads precanned demo data. The function call is LoadCannedData(). Table 2.15-24 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
Calls		
Function	**Where Described**	
StringToMapCoordinates	See Section 2.22.1.26.1.	
Called By		
Function	**Where Described**	
main	See Section 2.15.5.1.1.	

Table 2.15-24: LoadCannedData Information.

2.15.5.4 table.c

Development:SIMNET:MCC:Admin:table.c

table.c implements dialogs displaying tables of ammo and fuel supply trucks. Table 2.15-25 describes the variables used by table.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
ammoTruckTableDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
fuelTruckTableDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
ammoTruckData	TruckTableData	Development:SIMNET:MCC:Admin:table.c
fuelTruckData	TruckTableData	Development:SIMNET:MCC:Admin:table.c
ammoTruckTableColumns	array of FixTableColumnDefn	Development:SIMNET:libmac:table.h
ammoTruckTableField	FixTableFieldDefn	Development:SIMNET:libmac:table.h
byAmmoType	char	Standard C type.
ammoTruckLoadFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
ammoTruckTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
ammoTruckTableDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h
ammoTruckTableDialogState	pointer to DialogState	Development:SIMNET:libmac:dialog.h
fuelTruckTableColumns	array of FixTableColumnDefn	Development:SIMNET:libmac:table.h
fuelTruckTableField	FixTableFieldDefn	Development:SIMNET:libmac:table.h
fuelTruckTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fuelTruckTableDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h
fuelTruckTableDialogState	pointer to DialogState	Development:SIMNET:libmac:dialog.h

Table 2.15-25: table.c Variable Information.

2.15.5.4.1 SetUpTruckTables

SetUpTruckTables initializes the Ammo Truck and the Fuel Truck dialogs. The function call is SetUpTruckTables(). Table 2.15-26 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh
ShowDialog	See Section 2.22.1.11.1.

Called By	
Function	**Where Described**
SetUp	See Section 2.15.5.4.1.

Table 2.15-26: SetUpTruckTables Information.

2.15.5.4.2 ShowTruckTables

ShowTruckTables makes the Ammo Truck and Fuel Truck dialogs visible. The function call is ShowTruckTables(). Table 2.15-27 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
main	See Section 2.15.5.1.1.

Table 2.15-27: ShowTruckTables Information.

2.15.5.4.3 TruckTableFetch

TruckTableFetch initializes a truck table dialog, *dialog*, and updates the Dispatch and Load buttons. The function call is TruckTableFetch(dialog). Table 2.15-28 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
UpdateButtons	See Section 2.15.5.4.15.	

Table 2.15-28: TruckTableFetch Information.

2.15.5.4.4 TruckTableEvent

TruckTableEvent handles an event in a truck table dialog. The item number of the event is returned. The function call is TruckTableEvent(dialog, itemNo). Table 2.15-29 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
table	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
data	pointer to TruckTableData	Development:SIMNET:MCC: Admin:table.c
truck	pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
i	int	Standard C type.
rect	Rect	Development:THINK C: Mac #includes:MacTypes.h
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h

Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	the resource id of the button pressed.

Calls	
Function	**Where Described**
SelectWindow	Standard Window Manager function for Macintosh.
FixTableRowRect	See Section 2.22.1.42.5.
ShowHaltDialog	See Section 2.15.5.5.1.
ShowDispatchDialog	See Section 2.15.3.1.1.
ShowLoadDialog	See Section 2.15.1.1.1.
GetDItem	Standard Dialog Manager function for Macintosh.
GetPort	Standard Quickdraw function for Macintosh.
SetPort	Standard Quickdraw function for Macintosh.
InvalRect	Standard Dialog Manager function for Macintosh.

Table 2.15-29: TruckTableEvent Information.

2.15.5.4.5 TruckTableSelect

TruckTableSelect handles a mouse click in a table of trucks. The function call is TruckTableSelect(defn, row, box). Table 2.15-30 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
buf	register pointer to TruckTableData	Development:SIMNET:MCC: Admin:table.c
r	Rect	Development:THINK C: Mac #includes:MacTypes.h
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h

Calls	
Function	**Where Described**
GetPort	Standard Quickdraw function for Macintosh.
SetPort	Standard Quickdraw function for Macintosh.
FixTableRowRect	See Section 2.22.1.42.5.
InvertRect	Standard Quickdraw function for Macintosh.
UpdateBu:tons	See Section 2.15.5.4.15.

Table 2.15-30: TruckTableSelect Information.

2.15.5.4.6 TruckDrawNumber

TruckDrawNumber fills in the Truck Number column for the specified truck in the Truck Table. The function call is TruckDrawNumber(defn, col, row). Table 2.15-31 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
str	array of 10 char	Standard C type.

Calls	
Function	**Where Described**
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.
Move	Standard Quickdraw function for Macintosh.
CharWidth	Standard Quickdraw function for Macintosh.
DrawString	Standard Quickdraw function for Macintosh.

Table 2.15-31: TruckDrawNumber Information.

2.15.5.4.7 TruckDrawAssignment

TruckDrawAssignment fills in the Company Assignment column for the specified truck in the Truck Table. The function call is TruckDrawAssignment(defn, col, row). Table 2.15-32 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
pt	Point	Development:THINK C: Mac #includes:MacTypes.h
forceID	char	Standard C type.
Calls		
Function	**Where Described**	
GetPen	Standard Quickdraw function for Macintosh.	
MoveTo	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	
DrawChar	Standard Quickdraw function for Macintosh.	

Table 2.15-32: TruckDrawAssignment Information.

2.15.5.4.8 TruckDrawAmmoLoad

TruckDrawAmmoLoad fills in the Ammo Load column for the specified truck in the Truck Table. The function call is TruckDrawAmmoLoad(table, column, row). Table 2.15-33 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
table	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
column	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
rect	Rect	Development:THINK C: Mac #includes:MacTypes.h

Calls	
Function	**Where Described**
FixTableRowRect	See Section 2.22.1.42.5.
DisplayAmmoLoadSummary	See Section 2.22.3.7.1.

Table 2.15-33: TruckDrawAmmoLoad Information.

2.15.5.4.9 TruckDrawFuelLoad

TruckDrawFuelLoad fills in the Fuel Load column for the specified truck in the Truck Table. The function call is TruckDrawFuelLoad(defn, col, row). Table 2.15-34 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac:table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac:table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
str	array of 10 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.15-34: TruckDrawFuelLoad Information.

2.15.5.4.10 TruckDrawStatus

TruckDrawStatus fills in the Status column for the specified truck in the Truck Table. The function call is TruckDrawStatus(defn, col, row). Table 2.15-35 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac:table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac:table.h
row	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
statusString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.15-35: TruckDrawStatus Information.

2.15.5.4.11 TruckDrawLocation

TruckDrawLocation fills in the Location column for the specified truck in the Truck Table. The function call is TruckDrawLocation(defn, col, row). Table 2.15-36 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac:table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac:table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
str	pointer to char	Standard C type.
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.15-36: TruckDrawLocation Information.

2.15.5.4.12 TruckDrawETA

TruckDrawETA fills in the Estimated Time of Arrival column for the specified truck in the Truck Table. The function call is TruckDrawETA(defn, col, row). Table 2.15-37 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac:table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac:table.h
row	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
DTGToString	See Section 2.22.1.15.1.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.15-37: TruckDrawETA Information.

2.15.5.4.13 TruckTableHilite

TruckTableHilite inverts the text of the specified row if it is selected. If the truck for that row is disabled or destroyed, the row is pickled. The function call is TruckTableHilite(defn, row, box). Table 2.15-38 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
pattern	pointer to Pattern	Development:THINK C: Mac #includes:Quickdraw.h
destroyedPattern	Pattern	Development:THINK C: Mac #includes:Quickdraw.h
disabledPattern	Pattern	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
PenMode	Standard Quickdraw function for Macintosh.	
PenPat	Standard Quickdraw function for Macintosh.	
PaintRect	Standard Quickdraw function for Macintosh.	
InvertRect	Standard Quickdraw function for Macintosh.	

Table 2.15-38: TruckTableHilite Information.

2.15.5.4.14 UpdateTruckTableRow

UpdateTruckTableRow updates the row displaying a particular truck, *truck*. If the updated row is the selected one, the buttons are also updated. The function call is UpdateTruckTableRow(truck). Table 2.15-39 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
UpdateFixTableRow	See Section 2.22.1.42.2.	
UpdateButtons	See Section 2.15.5.4.15.	
Called By		
Function	**Where Described**	
AmmoTruckLoadComplete	See Section 2.15.1.1.2.	
CloseLoadDialog	See Section 2.15.1.1.6.	
ProcessRequest	See Section 2.15.2.1.1.	
ProcessResponse	See Section 2.15.2.1.2.	
CloseDispatchDialog	See Section 2.15.3.1.6.	
CloseHaltDialog	See Section 2.15.5.5.6.	
UpdateTruckLocation	See Section 2.15.5.7.3.	
DisableTruck	See Section 2.15.5.7.4.	
EnableTruck	See Section 2.15.5.7.5.	

Table 2.15-39: UpdateTruckTableRow Information.

2.15.5.4.15 UpdateButtons

UpdateButtons labels and enables push buttons according to the status of the currently selected truck. The Load button is enabled if a truck at the supply depot has been selected. The Dispatch/Halt button is labeled according to the truck selected. The Dispatch/Halt button is enabled if an operable truck is selected. The function call is UpdateButtons(dialog). Table 2.15-40 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h

Calls	
Function	**Where Described**
EnableControl	See Section 2.22.1.7.2.
DisableControl	See Section 2.22.1.7.1.
GetDItem	Standard Dialog Manager function for Macintosh.
SetCTitle	Standard Control Manager function for Macintosh.

Called By	
Function	**Where Described**
TruckTableFetch	See Section 2.15.5.4.3.
TruckTableSelect	See Section 2.15.5.4.5.
UpdateTruckTableRow	See Section 2.15.5.4.14.

Table 2.15-40: UpdateButtons Information.

2.15.5.4.16 SelectTruckTableRow

SelectTruckTableRow returns the currently selected, visible truck table row. The function call is SelectTruckTableRow(). Table 2.15-41 describes the internal variables used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
data	pointer to TruckTableData	Development:SIMNET:MCC: Admin:table.c

Return Values		
Return Value	**Type**	**Meaning**
0	pointer to TruckStatus	Unsuccessful.
&data->trucks[data->selection]	pointer to TruckStatus	Status data of the selected truck row.

Table 2.15-41: SelectTruckTableRow Information.

2.15.5.5 halt.c
Development:SIMNET:MCC:Admin:halt.c

halt.c implements dialogs for halting supply trucks. Table 2.15-42 describes the variables used by halt.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
haltAmmoTableField	ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
ammoTruckHaltFields	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fuelTruckHaltFields	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
ammoTruckHaltDialog	DialogDefn	Development:SIMNET:libmac:dialog.h
fuelTruckHaltDialog	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.15-42: halt.c Variable Information.

2.15.5.5.1 ShowHaltDialog

ShowHaltDialog pops up a dialog for ordering a truck, *truck*, to stop travelling. The function call is ShowHaltDialog(truck, window, rect). Table 2.15-43 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
window	WindowPtr	Development:THINK C:Mac #includes:WindowMgr.h
rect	pointer to Rect	Development:THINK C:Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
TruckTableEvent	See Section 2.15.5.4.4.	

Table 2.15-43: ShowHaltDialog Information.

2.15.5.5.2 TruckHaltFetch

TruckHaltFetch loads the text fields into the Truck Halt dialog and displays the load onboard the truck. The function call is TruckHaltFetch(dialog). Table 2.15-44 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of char	Standard C type.
Calls		
Function	**Where Described**	
UpdateHaltLocation	See Section 2.15.5.5.4.	
SetText	See Section 2.22.1.11.8.	
DTGToString	See Section 2.22.1.15.1.	
UpdateHaltLoad	See Section 2.15.5.5.5.	

Table 2.15-44: TruckHaltFetch Information.

2.15.5.5.3 TruckHaltEvent

TruckHaltEvent is called to report the truck's travel as having been halted when the Halt button is pressed, and to close the Truck Halt dialog when the Cancel button is pressed. The function call is TruckHaltEvent(dialog, itemNo). Table 2.15-45 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Calls		
Function	**Where Described**	
ReportHalt	See Section 2.15.2.1.4.	
CloseHaltDialog	See Section 2.15.5.5.6.	

Table 2.15-45: TruckHaltEvent Information.

2.15.5.5.4 UpdateHaltLocation

UpdateHaltLocation updates the location of the truck, *truck*, shown in the Truck Halt dialog. The function call is UpdateHaltLocation(truck). Table 2.15-46 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 100 char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
TruckHaltFetch	See Section 2.15.5.5.2.	
UpdateTruckLocation	See Section 2.15.5.7.3.	

Table 2.15-46: UpdateHaltLocation Information.

2.15.5.5.5 UpdateHaltLoad

UpdateHaltLoad updates the load displayed in the Truck Halt dialog. The function call is UpdateHaltLoad(). Table 2.15-47 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
EmptyScrollTable	See Section 2.22.1.28.2.	
FillAmmunitionList	See Section 2.22.3.5.1.	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
TruckHaltFetch	See Section 2.15.5.5.2.	

Table 2.15-47: UpdateHaltLoad Information.

2.15.5.5.6 CloseHaltDialog

CloseHaltDialog is called to close the Truck Halt dialog. The routine discards any scrolling list of ammo quantities, zooms back down to the Truck Table dialog, and updates the halted truck's row in the Truck Table dialog. The function call is CloseHaltDialog(). Table 2.15-48 describes the functions called using this function.

Calls	
Function	**Where Described**
DisposeScrollTable	See Section 2.22.1.30.2.
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
ThrowDialog	See Section 2.22.1.11.3.
UpdateTruckTableRow	See Section 2.15.5.4.14.
Called By	
Function	**Where Described**
TruckHaltEvent	See Section 2.15.5.5.3.

Table 2.15-48: CloseHaltDialog Information.

2.15.5.6 data.c

Development:SIMNET:MCC:Admin:data.c

data.c contains data definitions for the Admin/Log application. Table 2.15-49 describes the variables used by data.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
application	pointer to char	Standard C type.
authors	pointer to char	Standard C type.
copyright	pointer to char	Standard C type.
ammoTrucks	array of numberBnM977Vehicles TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
fuelTrucks	array of numberBnM978Vehicles TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
ammoSupplyPoint	MapCoordinates	Development:SIMNET:libmac: map.h
ammoSupplyContent	Manifest	Development:SIMNET:MCC: include:resupply.h
fuelSupplyPoint	MapCoordinates	Development:SIMNET:libmac: map.h
currentTruck	pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
zoomBaseWindow	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
zoomBaseRect	Rect	Development:THINK C: Mac #includes:MacTypes.h
popUpDialogState	pointer to DialogState	Development:SIMNET:libmac: dialog.h
(updatePopUpDialog)()	void function call	Standard
(closePopUpDialog)()	void function call	Standard
battleScheme	char	Standard C type.
bnForceID	char	Standard C type.
companyForceID	array of maxNumberCompanies char	Standard C type.
startupComplete	char	Standard C type.

Table 2.15-49: data.c Variable Information.

2.15.5.7 models.c

Development:SIMNET:MCC:Admin:model.c

models.c contains functions which model time-based processes involving supply trucks, such as breakdowns.

2.15.5.7.1 InitTruckState

InitTruckState initializes the state of a truck, *truck*. The function call is InitTruckState(truck). Table 2.15-50 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h

Calls	
Function	**Where Described**
Random	Standard Quickdraw function for Macintosh.
DisableTruck	See Section 2.15.5.7.4.
ScheduleDisableEvent	See Section 2.15.5.7.6.
Called By	
Function	**Where Described**
ProcessRequest	See Section 2.15.2.1.1.
SetUp	See Section 2.15.5.2.1.

Table 2.15-50: InitTruckState Information.

2.15.5.7.2 CheckTimers

CheckTimers looks for a time-driven change in status. The routine performs time-based processing once per minute, and checks for ammo and fuel trucks having moved, arrived, become broken or fixed. The function call is CheckTimers(). Table 2.15-51 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
now	unsigned long	Standard C type.
i	register short	Standard C type.
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
nextUpdateTime	unsigned long	Standard C type.
Calls		
Function	**Where Described**	
GetDateTime	Standard Operating System Utility function for Macintosh.	
EnableTruck	See Section 2.15.5.7.5.	
UpdateTruckLocation	See Section 2.15.5.7.3.	
DisableTruck	See Section 2.15.5.7.4.	
Called By		
Function	**Where Described**	
MainEventLoop	See Section 2.15.5.1.3.	

Table 2.15-51: CheckTimers Information.

2.15.5.7.3 UpdateTruckLocation

UpdateTruckLocation updates the status of a displacing truck. If the truck has arrived at its destination, any pop-up halt dialogs are discarded. If the truck's destination is its supply depot, the host and user are notified that the truck is ready for loading. If the truck has not yet arrived at its destination, the routine periodically updates an interpolation of its current location. The function call is UpdateTruckLocation(now, truck). Table 2.15-52 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
now	unsigned long	Standard C type.
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pt	pointer to LongPt	Development:SIMNET:libmac: longpt.h
str[100]	char	Standard C type.
Calls		
Function	**Where Described**	
DscardPopUpDialog	See Section 2.15.5.7.9.	
UpdateTruckTableRow	See Section 2.15.5.4.14.	
ReportHalt	See Section 2.15.2.1.4.	
NotifyStatusChange	See Section 2.15.5.7.7.	
InterpolatePoints	See Section 2.22.1.23.4.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
UpdateHaltLocation	See Section 2.15.5.5.4.	
Called By		
Function	**Where Described**	
CheckTimers	See Section 2.15.5.7.2.	

Table 2.15-52: UpdateTruckLocation Information.

2.15.5.7.4 DisableTruck

DisableTruck marks a truck, *truck*, as temporarily disabled. The routine first ensures that no more than 20% of the trucks in a fleet are disabled. Then, this truck is marked as disabled. Any current dialog pertaining to this truck is taken down. If the truck was travelling, the host and user are notified that it has stopped. The repair of the vehicle is scheduled. The function call is DisableTruck(truck). Table 2.15-53 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
fleetSize	int	Standard C type.
numberDisabled	int	Standard C type.
i	register short	Standard C type.
tp	register pointer to TruckStatus	Development:SIMNET:MCC: Admin:AdminMac.h

Calls	
Function	**Where Described**
DiscardPopUpDialog	See Section 2.22.6.5.9.
ReportHalt	See Section 2.15.2.1.4.
NotifyStatusChange	See Section 2.15.5.7.7.
UpdateTruckTableRow	See Section 2.15.5.4.14.
ScheduleDisableEvent	See Section 2.15.5.7.6.
Called By	
Function	**Where Described**
InitTruckState	See Section 2.15.5.7.1.
CheckTimers	See Section 2.15.5.7.2.

Table 2.15-53: DisableTruck Information.

2.15.5.7.5 EnableTruck

EnableTruck reenables a truck, *truck*, when it has been repaired. The user is notified that the truck is no longer disabled. The function call is EnableTruck(truck). Table 2.15-54 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
Calls		
Function	**Where Described**	
NotifyStatusChange	See Section 2.15.5.7.7.	
UpdateTruckTableRow	See Section 2.15.5.4.14.	
ScheduleDisableEvent	See Section 2.15.5.7.6.	
Called By		
Function	**Where Described**	
CheckTimers	See Section 2.15.5.7.2.	

Table 2.15-54: EnableTruck Information.

2.15.5.7.6 ScheduleDisableEvent

ScheduleDisableEvent determines the amount of time before a truck next becomes broken or fixed. The function call is ScheduleDisableEvent(truck, diceRange, numberDie). Table 2.15-55 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
diceRange	int	Standard C type.
numberDie	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dice	register short	Standard C type.
i	register short	Standard C type.
Calls		
Function	**Where Described**	
Random	Standard Quickdraw function for Macintosh.	
Called By		
Function	**Where Described**	
InitTruckState	See Section 2.15.5.7.1.	
DisableTruck	See Section 2.15.5.7.4.	
EnableTruck	See Section 2.15.5.7.5.	

Table 2.15-55: ScheduleDisableEvent Information.

2.15.5.7.7 NotifyStatusChange

NotifyStatusChange notifies the user of a change in the state of a truck, *truck*. *state* is the curent state of the truck. The function call is NotifyStatusChange(truck, state). Table 2.15-56 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
state	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cp	pointer to char	Standard C type.
str	array of 100 char	Standard C type.
Calls		
Function	**Where Described**	
CountCaution	See Section 2.22.1.4.4.	
DisposDialog	Standard Dialog Manager function for Macintosh.	
LastCaution	See Section 2.22.1.4.3.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.15.2.1.1.	
UpdateTruckLocation	See Section 2.15.5.7.3.	
DisableTruck	See Section 2.15.5.7.4.	
EnableTruck	See Section 2.15.5.7.5.	

Table 2.15-56: NotifyStatusChange Information.

2.15.5.7.8 UpdatePopUpLoad

UpdatePopUpLoad updates the load displayed in any pop-up dialog for the specified truck, *truck*. The function call is UpdatePopUpLoad(truck). Table 2.15-57 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.15.2.1.1.	

Table 2.15-57: UpdatePopUpLoad Information.

2.15.5.7.9 DiscardPopUpDialog

DiscardPopUpDialog discards any pop-up dialog for dispatching, halting or loading the specified truck, *truck*. The function call is DiscardPopUpDialog(truck). Table 2.15-58 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
truck	register pointer to TruckStatus	Development:SIMNET:MCC:Admin:AdminMac.h
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.15.2.1.1.	
ProcessResponse	See Section 2.15.2.1.2.	
UpdateTruckLocation	See Section 2.15.5.7.3.	

Table 2.15-58: DiscardPopUpDialog Information.

2.15.6 Appearance of Admin User Interface

2.15.6.1 Admin Pictures

Development:SIMNET:MCC:Admin:Admin Pictures

Admin Pictures contains resources that determine the appearance of the Admin application's user interface.

2.16 The Maint (Maintenance) Console

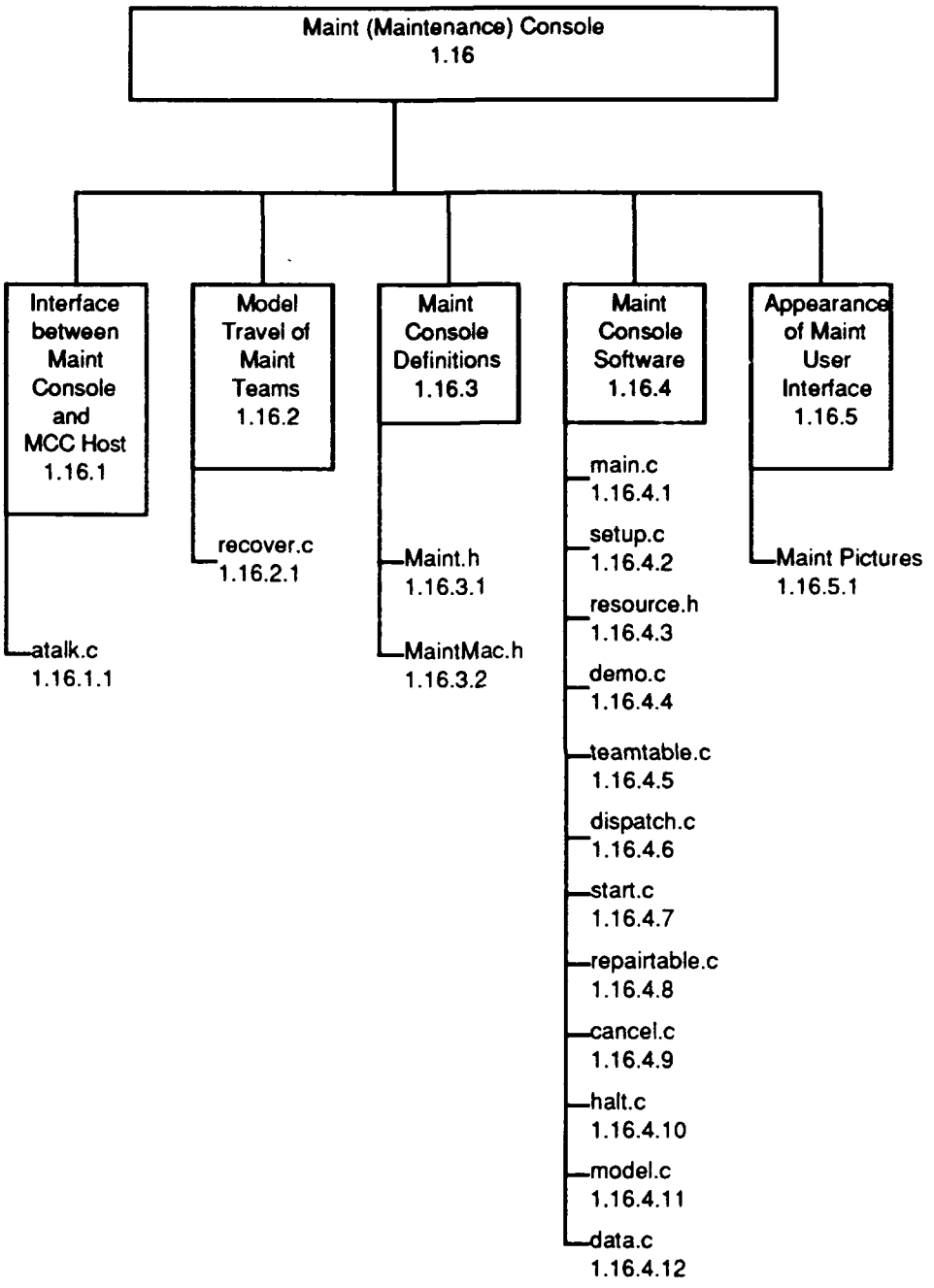


Figure 2.16-1: Maint (Maintenance) Console Structure.

2.16.1 Interface between Maint Console and MCC Host

2.16.1.1 atalk.c

Development:SIMNET:MCC:Maint:atalk.c

atalk.c processes requests and responses received from the MCC host via the AppleTalk network. This file is only compiled if VERSION is APPLETALK.

2.16.1.1.1 ProcessRequest

ProcessRequest processes a request received from the MCC host. This function is only compiled if VERSION is APPLETALK. The function call is ProcessRequest(hdl). Table 2.16-1 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
hdl	Handle	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	register Ptr	Development:THINK C: Mac #includes:MacTypes.h
errCode	int	Standard C type.
i	int	Standard C type.
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
repairSetP	pointer to pointer to RepairSet	Development:SIMNET:MCC: Maint:MaintMac.h
str	array of 100 char	Standard C type.
nameFormats	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
DiscardPopUpDialog	See Section 2.16.4.11.9.	
CancelRepair	See Section 2.16.4.9.5.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
InitTeamState	See Section 2.16.4.11.1.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	
ShowCaution	See Section 2.22.1.4.1.	
NotifyStateChange	See Section 2.16.4.11.8.	
ATPResponse	Standard Appletalk Manager function for Macintosh.	
Restart	Standard Operating System Utility function for Macintosh.	
SetDateTime	Standard Operating System Utility function for Macintosh.	
Called By		
Function	**Where Described**	
main	See Section 2.16.4.1.1.	

Table 2.16-1: ProcessRequest Information.

2.16.1.1.2 ProcessResponse

ProcessResponse processes a response received from the MCC host. This function is only compiled if VERSION is APPLETALK. The function call is ProcessResponse(hdl). Table 2.16-2 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
hdl	Handle	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	registerPtr	Development:THINK C: Mac #includes:MacTypes.h
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
Calls		
Function	**Where Described**	
PointToMapCoordinates	See Section 2.22.1.26.2.	
DiscardPopUpDialog	See Section 2.16.4.11.9.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	

Table 2.16-2: ProcessResponse Information.

2.16.1.1.3 ReportDispatch

ReportDispatch reports to the host that a team, *team*, has been dispatched. This function is only compiled if VERSION is APPLETALK. The function call is ReportDispatch(team). Table 2.16-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	MaintDispatchRequest	Development:SIMNET:MCC: Maint:Maint.h
Calls		
Function	**Where Described**	
ATPPut	See Section 2.22.1.3.3.	
Called By		
Function	**Where Described**	
TeamDispatchEvent	See Section 2.16.4.6.3.	
CheckTimers	See Section 2.16.4.11.2.	

Table 2.16-3: ReportDispatch Information.

2.16.1.1.4 ReportHalt

ReportHalt reports to the host that a team, *team*, has halted. This function is only compiled if VERSION is APPLETALK. The function call is ReportHalt(team). Table 2.16-4 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ab	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
req	pointer to MaintArriveRequest	Development:SIMNET:MCC: Maint:Maint.h
rsp	pointer to MaintArriveResponse	Development:SIMNET:MCC: Maint:Maint.h
i	register int	Standard C type.
Calls		
Function	**Where Described**	
SetUpATPRequest	See Section 2.22.1.3.2.	
ATPRequest	Standard Appletalk Manager function for Macintosh.	
Called By		
Function	**Where Described**	
TeamHaltEvent	See Section 2.16.4.10.3.	
UpdateTeamLocation	See Section 2.16.4.11.3.	
DisableTeam	See Section 2.16.4.11.5.	

Table 2.16-4: ReportHalt Information.

2.16.1.1.5 ReportRepair

ReportRepair reports to the host that a repair, *repair*, has been completed by *team*. The function call is ReportRepair(team, repair). Table 2.16-5 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
repair	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
req	MaintRepairRequest	Development:SIMNET:MCC: Maint:Maint.h

Calls	
Function	**Where Described**
ATPPut	See Section 2.22.1.3.3.
Called By	
Function	**Where Described**
RepairCompleted	See Section 2.16.4.11.4.

Table 2.16-5: ReportRepair Information.

2.16.2 Model Travel of Maint Teams

2.16.2.1 recover.c

Development:SIMNET:MCC:Maint:recover.c

recover.c implements a dialog for initiating towing of a vehicle by a maintenance team. Table 2.16-6 describes the variables used by recover.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
depotBuffer	char	Standard C type.
locationBuffer	MapCoordinates	Development:SIMNET:libmac: map.h
depotFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
recoverCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
locationField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
teamRecoverFields	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
teamRecoverDialog	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.16-6: recover.c Variable Information.

2.16.2.1.1 ShowRecoverDialog

ShowRecoverDialog pops up a dialog for ordering a team to recover. The function call is ShowRecoverDialog(team, window, rect). Table 2.16-7 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
rect	pointer to Rect	Development:THINK C: Mac #includes:MacTypes h

Calls	
Function	**Where Described**
ShowDialog	See Section 2.22.1.11.1.
ZoomToWindow	See Section 2.22.1.47.4.
ShowWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
TeamTableEvent	See Section 2.16.4.5.4.

Table 2.16-7: ShowRecoverDialog Information.

2.16.2.1.2 RecoverFetch

RecoverFetch reads the current values of the team recover data structures into the Recover dialog for display to the user. The function call is RecoverFetch(dialog). Table 2.16-8 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	

Table 2.16-8: RecoverFetch Information.

2.16.2.1.3 RecoverEvent

RecoverEvent handles an event in the Recover dialog. If the Recover button was pressed, the destination and arrival times are calculated. If the Cancel button is pressed, the dialog is closed. If the ComputeETA button is pressed, the arrival time is calculated and displayed. The function call is RecoverEvent(dialog, itemNo). Table 2.16-9 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 20 char	Standard C type.

Calls	
Function	**Where Described**
ComputeETA	See Section 2.16.2.1.4.
CloseRecoverDialog	See Section 2.16.2.1.5.
DTGToString	See Section 2.22.1.15.1.
SetText	See Section 2.22.1.11.8.
SellText	Standard Dialog Manager function for Macintosh.
SetRadioButton	See Section 2.22.1.11.6.

Table 2.16-9: RecoverEvent Information.

2.16.2.1.4 ComputeETA

ComputeETA computes the currently displayed team's arrival time. The function call is ComputeETA(). Table 2.16-10 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
distance	long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	No destination entered.
1	int	Successful.
Calls		
Function	**Where Described**	
ShowCaution	See Section 2.22.1.4.1.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
DistBetween2Pts	See Section 2.22.1.23.2.	
GetDateTime	Standard Operating System Utility function for Macintosh.	
DTGElapsed	See Section 2.22.1.15.3.	
Called By		
Function	**Where Described**	
RecoverEvent	See Section 2.16.2.1.3.	

Table 2.16-10: ComputeETA Information.

2.16.2.1.5 CloseRecoverDialog

CloseRecoverDialog is called to close the team recover dialog, *dialog*. The function call is CloseRecoverDialog(dialog). Table 2.16-11 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h

Calls	
Function	**Where Described**
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
ThrowDialog	See Section 2.22.1.11.3.
UpdateTeamTableRow	See Section 2.16.4.5.12.
Called By	
Function	**Where Described**
RecoverEvent	See Section 2.16.2.1.3.

Table 2.16-11: CloseRecoverDialog Information.

2.16.3 Maint Console Definitions

2.16.3.1 Maint.h
Development:SIMNET:MCC:Maint:Maint.h

Maint.h defines the representation of information communicated between the Maint Macintosh application and the MCC host.

2.16.3.2 MaintMac.h
Development:SIMNET:MCC:Maint:MaintMac.h

MaintMac.h defines types, constants and routines used within the Maint Macintosh application. Table 2.16-12 describes the variables used by MaintMac.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
repairClassName	extern pointer to array of char	Standard C type.
armorTeamCapabilities	extern TeamCapabilities	Development:SIMNET:MCC:Maint:MaintMac.h
repairTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
maintTeams	extern array of TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
selectedTeam	extern short	Standard C type.
currentTeam	extern pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
currentRepair	extern RepairHandle	Development:SIMNET:MCC:Maint:MaintMac.h
zoomBaseWindow	externWindowPtr	Development:THINK C:Mac #includes:WindowMgr.h
zoomBaseRect	extern Rect	Development:THINK C:Mac #includes:MacTypes.h
popUpDialogState	extern pointer to DialogState	Development:SIMNET:libmac:dialog.h
(closePopUpDialog)()	extern void function call	Standard
battleScheme	extern char	Standard C type.
bnForceID	extern char	Standard C type.
companyForceID	extern array of maxNumberCompanies char	Standard C type.
startupComplete	extern char	Standard C type.
repairDepot	extern MapCoordinates	Development:SIMNET:libmac:map.h

Table 2.16-12: MaintMac.h Variable Information.

2.16.4 Maint Console Software

2.16.4.1 main.c

Development:SIMNET:MCC:Maint:main.c

main.c contains the Maint application's program entry point and main event loop.

2.16.4.1.1 main

main is the application entry point. The function call is main(). Table 2.16-13 describes the functions called using this function.

Calls	
Function	**Where Described**
SetUp	See Section 2.16.4.2.1.
ProcessRequest	See Section 2.16.1.1.1.
SetUpAppleTalk	See Section 2.22.1.3.1.
DownloadTerrainMap	See Section 2.22.1.3.4.
DownloadInitialData	See Section 2.16.4.1.2.
LoadCannedData	See Section 2.16.4.4.1.
MenuBarTitle	See Section 2.22.1.43.1.
InstallClock	See Section 2.22.1.6.4.
ShowTeamTable	See Section 2.16.4.5.2.
MainEventLoop	See Section 2.16.4.1.3.

Table 2.16-13: main Information.

2.16.4.1.2 DownloadInitialData

DownloadInitialData downloads information about the maintenance teams and repairs depots. This function is only compiled if VERSION is APPLETALK. The function call is DownloadInitialData(). Table 2.16-14 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
myEvent	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.	
Called By		
Function	**Where Described**	
main	See Section 2.16.4.1.1.	

Table 2.16-14: DownloadInitialData Information.

2.16.4.1.3 MainEventLoop

MainEventLoop sits in a loop forever fielding incoming events. The function call is MainEventLoop(). Table 2.16-15 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
event	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
repair	RepairHandle	Development:SIMNET:MCC: Maint:MaintMac.h
str	array of 100 char	Standard C type.

Calls	
Function	**Where Described**
UpdateClock	See Section 2.22.1.6.2.
CheckTimers	See Section 2.16.4.11.2.
SystemTask	Standard Desk Manager function for Macintosh.
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
UpdateTeamTableRow	See Section 2.16.4.5.12.
ATPCloseSocket	Standard Appletalk Manager function for Macintosh.
ExitToShell	Standard Segment Loader function for Macintosh.
GetDateTime	Standard Operating System Utility function for Macintosh.
ShowVersions	See Section 2.22.1.44.1.
WindowEvent	See Section 2.22.1.46.2.
NetworkEventHandler	See Section 2.22.1.3.5.
Called By	
Function	**Where Described**
main	See Section 2.16.4.1.1.

Table 2.16-15: MainEventLoop Information.

2.16.4.2 setup.c

Development:SIMNET:MCC:Maint:setup.c

setup.c initializes the Maint application at start-up.

2.16.4.2.1 SetUp

SetUp initializes the Maint application. The SIMNET Resources and Maint Pictures files are opened. The random number generator is initialized, the table of team information is initialized, and the dialogs are initialized. The function call is SetUp(). Table 2.16-16 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
port	GrafPort	Development:THINK C: Mac #includes:Quickdraw.h
finalTicks	long	Standard C type.
i	register short	Standard C type.
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h

Calls	
Function	**Where Described**
InitToolbox	See Section 2.22.1.20.1.
ZoomInit	See Section 2.22.1.47.1.
OpenResFile	Standard Resource Manager function for Macintosh.
MenuBarTitle	See Section 2.22.1.43.1.
Delay	Standard Operating System Utility function for Macintosh.
ExitToShell	Standard Segment Loader function for Macintosh.
DeepShit	See Section 2.22.1.8.2.
GetDateTime	Standard Operating System Utility function for Macintosh.
InitTeamState	See Section 2.16.4.11.1.
NewPtr	Standard Memory Manager function for Macintosh.
OpenPort	Standard Quickdraw function for Macintosh.
SetUpRepairTable	See Section 2.16.4.8.1.
SetUpTeamTable	See Section 2.16.4.5.1.
ClosePort	Standard Quickdraw function for Macintosh.
Called By	
Function	**Where Described**
main	See Section 2.16.4.1.1.

Table 2.16-16: SetUp Information.

2.16.4.3 resource.h

Development:SIMNET:MCC:Maint:resource.h

resource.h defines the resource numbers of the Maint Pictures resources.

2.16.4.4 demo.c

Development:SIMNET:MCC:Maint:demo.c

demo.c contains a function which loads precanned values for the demo version of the Maint console application.

2.16.4.4.1 LoadCannedData

LoadCannedData loads precanned demo data. The function call is LoadCannedData(). Table 2.16-17 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
team	pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
Calls		
Function	**Where Described**	
StringToMapCoordinates	See Section 2.22.1.26.1.	
Called By		
Function	**Where Described**	
main	See Section 2.16.4.1.1.	

Table 2.16-17: LoadCannedData Information.

2.16.4.5 teamtable.c
Development:SIMNET:MCC:Maint:teamtable.c

teamtable.c implements a dialog displaying a table of maintenance teams. Table 2.16-18 describes the variables used by teamtable.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
teamTableColumns	array of FixTableColumnDefn	Development:SIMNET:libmac: table.h
teamTableField	FixTableFieldDefn	Development:SIMNET:libmac: table.h
teamTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
teamTableDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h
teamTableDialogState	pointer to DialogState	Development:SIMNET:libmac: dialog.h

Table 2.16-18: teamtable.c Variable Information.

2.16.4.5.1 SetUpTeamTable

SetUpTeamTable initializes a truck table dialog. The function call is SetUpTeamTable(). Table 2.16-19 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
NewPtr	Standard Memory Manager function for Macintosh.
ShowDialog	See Section 2.22.1.11.1.
Called By	
Function	**Where Described**
SetUp	See Section 2.16.4.2.1.

Table 2.16-19: SetUpTeamTable Information.

2.16.4.5.2 ShowTeamTable

ShowTeamTable makes the maintenance team table dialog visible. The function call is ShowTeamTable(). Table 2.16-20 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
SelectWindow	Standard Window Manager function for Macintosh.

Called By	
Function	**Where Described**
main	See Section 2.16.4.1.1.
ShowTeamTable	See Section 2.16.4.8.4.

Table 2.16-20: ShowTeamTable Information.

2.16.4.5.3 TeamTableFetch

TeamTableFetch initializes the maintenance team table dialog and updates the action buttons. The function call is TeamTableFetch(dialog). Table 2.16-21 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
UpdateTeamTableButtons	See Section 2.16.4.5.13.	

Table 2.16-21: TeamTableFetch Information.

2.16.4.5.4 TeamTableEvent

TeamTableEvent handles an event in the maintenance team table dialog. If the Repairs button is pressed, the Repair Table is displayed. If the Dispatch button is pressed, the appropriate Dispatch or Halt dialog is displayed. If the Recover button is pressed, the tow dialog is displayed. If the StartRepair button is pressed, the repair selection is displayed. The function call is TeamTableEvent(dialog, itemNo). Table 2.16-22 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
i	int	Standard C type.
rect	Rect	Development:THINK C: Mac #includes:MacTypes.h
Return Values		
Return Value	**Type**	**Meaning**
itemNo	int	

Calls	
Function	**Where Described**
ShowRepairTable	See Section 2.16.4.8.2.
FixTableRowRect	See Section 2.22.1.42.5.
ShowDispatchDialog	See Section 2.16.4.6.1.
ShowHaltDialog	See Section 2.16.4.10.1.
ShowRecoverDialog	See Section 2.16.2.1.1.
ShowStartDialog	See Section 2.16.4.7.1.

Table 2.16-22: TeamTableEvent Information.

2.16.4.5.5 TeamTableSelect

TeamTableSelect handles a mouse click in the table of maintenance teams. The function call is TeamTableSelect(defn, row, box). Table 2.16-23 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac:table.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
r	Rect	Development:THINK C: Mac #includes:MacTypes.h
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
FixTableRowRect	See Section 2.22.1.42.5.	
InvertRect	Standard Quickdraw function for Macintosh.	
UpdateTeamTableButtons	See Section 2.16.4.5.13.	

Table 2.16-23: TeamTableSelect Information.

2.16.4.5.6 TeamDrawNumber

TeamDrawNumber fills in the Team Number field of the specified column in the Team Table. The function call is TeamDrawNumber(defn, col, row). Table 2.16-24 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table .h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
str	array of 10 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
CharWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-24: TeamDrawNumber Information.

2.16.4.5.7 TeamDrawAssignment

TeamDrawAssignment fills in the Team Assignment field of the specified column in the Team Table. The function call is TeamDrawAssignment(defn, col, row). Table 2.16-25 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
pt	Point	Development:THINK C: Mac #includes:MacTypes.h
forceID	char	Standard C type.

Calls	
Function	**Where Described**
GetPen	Standard Quickdraw function for Macintosh.
MoveTo	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
DrawString	Standard Quickdraw function for Macintosh.
DrawChar	Standard Quickdraw function for Macintosh.

Table 2.16-25: TeamDrawAssignment Information.

2.16.4.5.8 TeamDrawStatus

TeamDrawStatus fills in the Team Status field of the specified column in the Team Table. The function call is TeamDrawStatus(defn, col, row). Table 2.16-26 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
statusString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.16-26: TeamDrawStatus Information.

2.16.4.5.9 TeamDrawLocation

TeamDrawLocation fills in the Team Location field of the specified column in the Team Table. The function call is TeamDrawLocation(defn, col, row). Table 2.16-27 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
str	pointer to char	Standard C type.
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.16-27: TeamDrawLocation Information.

2.16.4.5.10 TeamDrawETA

TeamDrawETA fills in the Team ETA field of the specified column in the Team Table. The function call is TeamDrawETA(defn, col, row). Table 2.16- 28 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDefn	Development:SIMNET:libmac: table.h
col	pointer to FixTableColumnDefn	Development:SIMNET:libmac: table.h
row	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
DTGToString	See Section 2.22.1.15.1.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-28: TeamDrawETA Information.

2.16.4.5.11 TeamTableHilite

TeamTableHilite inverts the text of the selected row so the user can see that the item is selected. If the selected team is disabled or destroyed, the row is pickled. The function call is TeamTableHilite(defn, row, box). Table 2.16-29 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to FixTableFieldDfen	Development:SIMNET:libmac: table.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
pattern	pointer to Pattern	Development:THINK C: Mac #includes:Quickdraw.h
destroyedPattern	Pattern	Development:THINK C: Mac #includes:Quickdraw.h
disabledPattern	Pattern	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
PenMode	Standard Quickdraw function for Macintosh.	
PenPat	Standard Quickdraw function for Macintosh.	
PaintRect	Standard Quickdraw function for Macintosh.	
InvertRect	Standard Quickdraw function for Macintosh.	

Table 2.16-29: TeamTableHilite Information.

2.16.4.5.12 UpdateTeamTableRow

UpdateTeamTableRow updates the row displaying a particular team, *team*. The function call is UpdateTeamTableRow(team). Table 2.16-30 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
Calls		
Function	**Where Described**	
UpdateFixTableRow	See Section 2.22.1.42.2.	
UpdateTeamTableButtons	See Section 2.16.4.5.13.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.16.1.1.1.	
ProcessResponse	See Section 2.16.1.1.2.	
CloseRecoverDialog	See Section 2.16.2.1.5.	
MainEventLoop	See Section 2.16.4.1.3.	
CloseDispatchDialog	See Section 2.16.4.6.5.	
CloseStartDialog	See Section 2.16.4.7.11.	
CancelRepair	See Section 2.16.4.9.5.	
CloseHaltDialog	See Section 2.16.4.10.5.	
CheckTimers	See Section 2.16.4.11.2.	
UpdateTeamLocation	See Section 2.16.4.11.3.	
RepairCompleted	See Section 2.16.4.11.4.	
DisableTeam	See Section 2.16.4.11.5.	
EnableTeam	See Section 2.16.4.11.6.	

Table 2.16-30: UpdateTeamTableRow Information.

2.16.4.5.13 UpdateTeamTableButtons

UpdateTeamTableButtons labels and enables push buttons according to the status of the currently selected team. The function call is UpdateTeamTableButtons(). Table 2.16-31 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
enable	char	Standard C type.
repair	register short	Standard C type.

Calls	
Function	**Where Described**
GetDItem	Standard Dialog Manager function for Macintosh.
SetCTitle	Standard Control Manager function for Macintosh.
EnableControl	See Section 2.22.1.7.2.
DisableControl	See Section 2.22.1.7.1.

Called By	
Function	**Where Described**
TeamTableFetch	See Section 2.16.4.5.3.
TeamTableSelect	See Section 2.16.4.5.5.
UpdateTeamTableRow	See Section 2.16.4.5.12.

Table 2.16-31: UpdateTeamTableButtons Information.

2.16.4.6 dispatch.c
Development:SIMNET:MCC:Maint:dispatch.c

dispatch.c implements a dialog for dispatching a maintenance team. Table 2.16-32 describes the variables used by dispatch.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
assignmentBuffer	char	Standard C type.
depotBuffer	char	Standard C type.
locationBuffer	MapCoordinates	Development:SIMNET:libmac:map.h
assignmentFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
depotFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
dispatchCancelField	PushButtonFieldDefn	Development:SIMNET:libmac:dialog.h
locationField	CoordinatesFieldDefn	Development:SIMNET:libmac:dialog.h
teamDispatchFields	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
teamDispatchDialog	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.16-32: dispatch.c Variable Information.

2.16.4.6.1 ShowDispatchDialog

ShowDispatchDialog pops up a dialog for ordering a team to dispatch. The function call is ShowDispatchDialog(team, window, rect). Table 2.16-33 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
window	WindowPtr	Development:THINK C:Mac #includes:WindowMgr.h
rect	pointer to Rect	Development:THINK C:Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
TeamTableEvent	See Section 2.16.4.5.4.	

Table 2.16-33: ShowDispatchDialog Information.

2.16.4.6.2 TeamDispatchFetch

TeamDispatchFetch reads the current values of the data structures describing the Team Dispatch dialog into the dialog for display to the user. The function call is TeamDispatchFetch(dialog). Table 2.16-34 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
str	array of 50 char	Standard C type.
forceID	char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
DisableControl	See Section 2.22.1.7.1.	

Table 2.16-34: TeamDispatchFetch Information.

2.16.4.6.3 TeamDispatchEvent

TeamDispatchEvent handles events for the Team *Dispatch dialog.* If the Dispatch button is pressed, the destination and arrival times are computed, the radio buttons are updated, and the team is reported as dispatched. If the Cancel button is pressed, the dialog is closed. If the ComputeETA button is pressed, the estimated arrival time is calculated. The function call is TeamDispatchEvent(dialog, itemNo). Table 2.16-35 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 20 char	Standard C type.

Calls	
Function	**Where Described**
ComputeETA	See Section 2.16.4.6.4.
ReportDispatch	See Section 2.16.1.1.3.
CloseDispatchDialog	See Section 2.16.4.6.5.
DTGToString	See Section 2.22.1.15.1.
SetText	See Section 2.22.1.11.8.
SellText	Standard Dialog Manager function for Macintosh.
SetRadioButton	See Section 2.22.1.11.6.

Table 2.16-35: TeamDispatchEvent Information.

2.16.4.6.4 ComputeETA

ComputeETA computes the currently displayed team's arrival time. The function call is ComputeETA(). Table 2.16-36 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
distance	long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	No destination entered.
1	int	Successful.
Calls		
Function	**Where Described**	
ShowCaution	See Section 2.22.1.4.1.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
DistBetween2Pts	See Section 2.22.1.23.2.	
GetDateTime	Standard Operating System Utility function for Macintosh.	
DTGElapsed	See Section 2.22.1.15.3.	
Called By		
Function	**Where Described**	
TeamDispatchEvent	See Section 2.16.4.6.3.	

Table 2.16-36: ComputeETA Information.

2.16.4.6.5 CloseDispatchDialog

CloseDispatchDialog is called to close the team dispatch dialog. The function call is CloseDispatchDialog(dialog). Table 2.16-37 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h

Calls	
Function	**Where Described**
HideWindow	Standard Window Manager function for Macintosh.
ZoomToWindow	See Section 2.22.1.47.4.
ThrowDialog	See Section 2.22.1.11.3.
UpdateTeamTableRow	See Section 2.16.4.5.12.
Called By	
Function	**Where Described**
TeamDispatchEVent	See Section 2.16.4.6.3.

Table 2.16-37: CloseDispatchDialog Information.

2.16.4.7 start.c

Development:SIMNET:MCC:Maint:start.c

start.c implements a dialog used to select and start a repair. Table 2.16-38 describes the variables used by start.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
startColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
startTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
startFields	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
startDialog	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.16-38: start.c Variable Information.

2.16.4.7.1 ShowStartDialog

ShowStartDialog displays the dialog for selecting and starting a repair. The function call is ShowStartDialog(team, window, rect). Table 2.16-39 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	pointer to TeamStatus	Development:SIMNET:MCC. Maint:MaintMac.h
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
rect	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for Macintosh.	

Called By	
Function	**Where Described**
TeamTableEvent	See Section 2.16.4.5.4.

Table 2.16-39: ShowStartDialog Information.

2.16.4.7.2 StartFetch

StartFetch reads the repair types table into the Start dialog for display to the user. The function call is StartFetch(dialog). Table 2.16-40 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
repairSet	register pointer to RepairSet	Development:SIMNET:MCC:Maint:MaintMac.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac:scroll.h
i	register short	Standard C type.
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
NewHandle	Standard Memory Manager function for Macintosh.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
DisableControl	See Section 2.22.1.7.1.	

Table 2.16-40: StartFetch Information.

2.16.4.7.3 StartEvent

StartEvent handles events in the Start dialog. If the Start button is pressed, a new repair is started. If the Cancel button is pressed, the Start dialog is closed. The function call is StartEvent(dialog, itemNo). Table 2.16-41 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Calls		
Function	**Where Described**	
NewRepair	See Section 2.16.4.7.12.	
CloseStartDialog	See Section 2.16.4.7.11.	

Table 2.16-41: StartEvent Information.

2.16.4.7.4 StartSelect

StartSelect handles a mouse click in the table of repair types. The function call is StartSelect(defn, row, box, event). Table 2.16-42 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
SelectScrollTableEntry	See Section 2.22.1.37.1.	
UpdateStartButton	See Section 2.16.4.7.10.	

Table 2.16-42: StartSelect Information.

2.16.4.7.5 StartHilite

StartHilite hilites a row in the table of repair types. The function call is StartHilite(defn, entry, box). Table 2.16-43 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
unavailablePattern	Pattern	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
PenMode	Standard Quickdraw function for Macintosh.	
PenPat	Standard Quickdraw function for Macintosh.	
PaintRect	Standard Quickdraw function for Macintosh.	
InvertRect	Standard Quickdraw function for Macintosh.	

Table 2.16-43: StartHilite Information.

2.16.4.7.6 StartDrawDescription

StartDrawDescription fills in the Description field of the specified column in the Start Table. The function call is StartDrawDescription(defn, col, entry, box). Table 2.16-44 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:macTypes.h
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-44: StartDrawDescription Information.

2.16.4.7.7 StartDrawClass

StartDrawClass fills in the Class field of the specified column in the Start Table. The function call is StartDrawClass(defn, col, entry, box). Table 2.16-45 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-45: StartDrawClass Information.

2.16.4.7.8 StartDrawDuration

StartDrawDuration fills in the Duration field of the specified column in the Start Table. The function call is StartDrawDuration(defn, col, entry, box). Table 2.16-46 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-46: StartDrawDuration Information.

2.16.4.7.9 UpdateStartDialog

UpdateStartDialog updates the choice of repairs in the Start dialog when a repair is completed. The function call is UpdateStartDialog(). Table 2.16-47 describes the internal parameters used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
UpdateStartButton	See Section 2.16.4.7.10.	
Called By		
Function	**Where Described**	
RepairCompleted	See Section 2.16.4.11.4.	

Table 2.16-47: UpdateStartDialog Information.

2.16.4.7.10 UpdateStartButton

UpdateStartButton enables or disables the StartRepair button. The button is enabled if and only if a repair type is selected and no repairs of the same class as the selected repair are currently being performed by the subject team. The function call is UpdateStartButton(). Table 2.16-48 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
repair	register pointer to RepairDescriptor	Development:SIMNET:MCC:Maint:MaintMac.h
Calls		
Function	**Where Described**	
EnableControl	See Section 2.22.1.7.2.	
DisableControl	See Section 2.22.1.7.1.	
Called By		
Function	**Where Described**	
StartSelect	See Section 2.16.4.7.4.	
UpdateStartDialog	See Section 2.16.4.7.9.	

Table 2.16-48: UpdateStartButton Information.

2.16.4.7.11 CloseStartDialog

CloseStartDialog is called to close the repair selection dialog, *dialog*. The function call is CloseStartDialog(dialog). Table 2.16-49 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomToWindow	See Section 2.22.1.47.4.	
ThrowDialog	See Section 2.22.1.11.3.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	
Called By		
Function	**Where Described**	
StartEvent	See Section 2.16.4.7.3.	

Table 2.16-49: CloseStartDialog Information.

2.16.4.7.12 NewRepair

NewRepair commences a new repair, *desc*, with *team*. The function call is NewRepair(team, desc). Table 2.16-50 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
desc	pointer to RepairDescriptor	Development:SIMNET:MCC:Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
repair	register RepairHandle	Development:SIMNET:MCC:Maint:MaintMac.h
repairSerialNumber	int	Standard C type.
now	unsigned long	Standard C type.
dtg	DateTimeGroup	Development:SIMNET:libmac:dtg.h
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
GetDateTime	Standard Operating System Utility function for Macintosh.	
DTGElapsed	See Section 2.22.1.15.3.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
Called By		
Function	**Where Described**	
StartEvent	See Section 2.16.4.7.3.	

Table 2.16-50: NewRepair Information.

2.16.4.8 repairtable.c

Development:SIMNET:MCC:Maint:repairtable.c

repairtable.c implements a dialog displaying a scrolling table of repairs. Table 2.16-51 describes the variables used by repairtable.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
repairTableColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
repairTableField	ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
repairTableFieldlist	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
repairTableDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h
repairTableDialogState	pointer to DialogState	Development:SIMNET:libmac: dialog.h

Table 2.16-51: repairtable.c Variable Information.

2.16.4.8.1 SetUpRepairTable

SetUpRepairTable initializes the repair table dialog. The function call is SetUpRepairTable(). Table 2.16-52 describes the functions called using this function.

Calls	
Function	**Where Described**
TextFont	Standard Quickdraw function for Macintosh.
TextSize	Standard Quickdraw function for Macintosh.
StringWidth	Standard Quickdraw function for Macintosh.
NewPtr	Standard Memory Manager function for Macintosh.
ShowDialog	See Section 2.22.1.11.1.
Called By	
Function	**Where Described**
SetUp	See Section 2.16.4.2.1.

Table 2.16-52: SetUpRepairTable Information.

2.16.4.8.2 ShowRepairTable

ShowRepairTable makes the repair table dialog visible. The function call is ShowRepairTable(). Table 2.16-53 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
SelectWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
TeamTableEvent	See Section 2.16.4.5.4.

Table 2.16-53: ShowRepairTable Information.

2.16.4.8.3 RepairTableFetch

RepairTableFetch initializes the repair table dialog, *dialog*. The function call is RepairTableFetch(dialog). Table 2.16-54 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
DisableControl	See Section 2.22.1.7.1.	

Table 2.16-54: RepairTableFetch Information.

2.16.4.8.4 RepairTableEvent

RepairTableEvent handles an event in the repair table dialog. The function call is RepairTableEvent(dialog, itemNo). Table 2.16-55 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Calls		
Function	**Where Described**	
ShowTeamTable	See Section 2.16.4.5.2.	
ShowCancelDialog	See Section 2.16.4.9.1.	

Table 2.16-55: RepairTableEvent Information.

2.16.4.8.5 RepairTableSelect

RepairTableSelect handles a mouse click in the table of repairs in progress. The function call is RepairTableSelect(defn, row, box, event). Table 2.16-56 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h

Calls	
Function	**Where Described**
SelectScrollTableEntry	See Section 2.22.1.37.1.
EnableControl	See Section 2.22.1.7.2.
DisableControl	See Section 2.22.1.7.1.

Table 2.16-56: RepairTableSelect Information.

2.16.4.8.6 RepairTableDrawVehicle

RepairTableDrawVehicle fills in the Vehicle field of the specified column in the Repair Table. The function call is RepairTableDrawVehicle(defn, col, entry, box). Table 2.16-57 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-57: RepairTableDrawVehicle Information.

2.16.4.8.7 RepairTableDrawLocation

RepairTableDrawLocation fills in the Location field of the specified column in the Repair Table. The function call is RepairTableDrawLocation(defn, col, entry, box). Table 2.16-58 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.16-58: RepairTableDrawLocation Information.

2.16.4.8.8 RepairTableDrawDisabled

RepairTableDrawDisabled fills in the Disabled field of the specified column in the Repair Table. The function call is RepairTableDrawDisabled(defn, col, entry, box). Table 2.16-59 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
r	Rect	Development:THINK C: Mac #includes:MacTypes.h
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
TextBox	Standard TextEdit function for Macintosh.	

Table 2.16-59: RepairTableDrawDisabled Information.

2.16.4.8.9 RepairTableDrawDescription

RepairTableDrawDescription fills in the Description field of the specified column in the Repair Table. The function call is RepairTableDrawDescription(defn, col, entry, box). Table 2.16-60 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:macTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cp	pointer to char	Standard C type.
r	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
TextBox	Standard TextEdit function for Macintosh.	

Table 2.16-60: RepairTableDrawDescription Information.

2.16.4.8.10 RepairTableDrawETC

RepairTableDrawETC fills in the ETC field of the specified column in the Repair Table. The function call is RepairTableDrawETC(defn, col, entry, box). Table 2.16-61 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
DTGToString	See Section 2.22.1.15.1.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-61: RepairTableDrawETC Information.

2.16.4.8.11 RepairTableDrawState

RepairTableDrawState fills in the State field of the specified column in the Repair Table. The function call is RepairTableDrawState(defn, col, entry, box). Table 2.16-62 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
stateString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.16-62: RepairTableDrawState Information.

2.16.4.8.12 UpdateRepairTableRow

UpdateRepairTableRow updates the row displaying a particular repair, *repair*. The function call is UpdateRepairTableRow(repair). Table 2.16-63 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
repair	register RepairHandle	Development:SIMNET:MCC: Maint:MaintMac.h
Calls		
Function	**Where Described**	
UpdateScrollTableEntry	See Section 2.22.1.31.1.	
EnableControl	See Section 2.22.1.7.2.	
DisableControl	See Section 2.22.1.7.1.	
Called By		
Function	**Where Described**	
CancelRepair	See Section 2.16.4.9.5.	
RepairCompleted	See Section 2.16.4.11.4.	

Table 2.16-63: UpdateRepairTableRow Information.

2.16.4.9 cancel.c
Development:SIMNET:MCC:Maint:cancel.c

cancel.c implements a dialog for cancelling a repair in progress. Table 2.16-64 describes the variables used by cancel.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
repairCancelFields	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
repairCancelDialog	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.16-64: cancel.c Variable Information.

2.16.4.9.1 ShowCancelDialog

ShowCancelDialog pops up the dialog for ordering a repair vehicle to stop travelling. The function call is ShowCancelDialog(). Table 2.16-65 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowDialog	See Section 2.22.1.11.1.
ZoomScrollTableEntry	See Section 2.22.1.36.1.
Called By	
Function	**Where Described**
RepairTableEvent	See Section 2.16.4.8.4.

Table 2.16-65: ShowCancelDialog Information.

2.16.4.9.2 RepairCancelFetch

RepairCancelFetch reads the text fields into the Repair Cancel dialog. The function call is RepairCancelFetch(dialog). Table 2.16-66 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
str	array of 50 char	Standard C type.

Calls	
Function	**Where Described**
SetText	See Section 2.22.1.11.8.
GetDItem	Standard Dialog Manager function for Macintosh.
SetIText	Standard Dialog Manager function for Macintosh.
DTGToString	See Section 2.22.1.15.1.

Table 2.16-66: RepairCancelFetch Information.

2.16.4.9.3 RepairCancelEvent

RepairCancelEvent handles events in the Repair Cancel dialog. If the Cancel button is pressed, the repair is cancelled. If the DontCancel button is pressed, the Cancel dialog is closed. The function call is RepairCancelEvent(dialog, itemNo). Table 2.16-67 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
Calls		
Function	**Where Described**	
CancelRepair	See Section 2.16.4.9.5.	
CloseCancelDialog	See Section 2.16.4.9.4.	

Table 2.16-67: RepairCancelEvent Information.

2.16.4.9.4 CloseCancelDialog

CloseCancelDialog is called to close the repair cancel dialog, *dialog*. The function call is CloseCancelDialog(dialog). Table 2.16-68 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Calls		
Function	**Where Described**	
ZoomScrollTableEntry	See Section 2.22.1.36.1.	
ThrowDialog	See Section 2.22.1.11.3.	

Called By	
Function	**Where Described**
RepairCancelEvent	See Section 2.16.4.9.3.

Table 2.16-68: CloseCancelDialog Information.

2.16.4.9.5 CancelRepair

CancelRepair cancels a repair, *repair*, in progress. The function call is CancelRepair(repair). Table 2.16-69 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
repair	register RepairHandle	Development:SIMNET:MCC:Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
Calls		
Function	**Where Described**	
UpdateRepairTableRow	See Section 2.16.4.8.12.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.16.1.1.1.	
RepairCancelEvent	See Section 2.16.4.9.3.	

Table 2.16-69: CancelRepair Information.

2.16.4.10 halt.c

Development:SIMNET:MCC:Maint:halt.c

halt.c implements a dialog for halting a maintenance team. Table 2.16-70 describes the varibles used by halt.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
teamHaltFields	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
teamHaltDialog	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.16-70: halt.c Variable Information.

2.16.4.10.1 ShowHaltDialog

ShowHaltDialog pops up a dialog for ordering a team to stop travelling. The function call is ShowHaltDialog(team, window, rect). Table 2.16-71 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
rect	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ZoomToWindow	See Section 2.22.1.47.4.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
TeamTableEvent	See Section 2.16.4.5.4.	

Table 2.16-71: ShowHaltDialog Information.

2.16.4.10.2 TeamHaltFetch

TeamHaltFetch loads the text fields into the Team Halt dialog. The function call is TeamHaltFetch(dialog). Table 2.16-72 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 50 char	Standard C type.
stateString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
UpdateHaltLocation	See Section 2.16.4.10.4.	
SetText	See Section 2.22.1.11.8.	
DTGToString	See Section 2.22.1.15.1.	

Table 2.16-72: TeamHaltFetch Information.

2.16.4.10.3 TeamHaltEvent

TeamHaltEvent handles events in the Team Halt dialog. If the Halt button is pressed, the routine reports that the team travel has halted. If the Cancel button is pressed, the dialog is closed. The function call is TeamHaltEvent(dialog, itemNo). Table 2.16-73 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Calls		
Function	**Where Described**	
ReportHalt	See Section 2.16.1.1.4.	
CloseHaltDialog	See Section 2.16.4.10.5.	

Table 2.16-73: TeamHaltEvent Information.

2.16.4.10.4 UpdateHaltLocation

UpdateHaltLocation updates the location of the team, *team*, shown in the halt dialog. The function call is UpdateHaltLocation(team). Table 2.16-74 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 50 char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
TeamHaltFetch	See Section 2.16.4.10.2.	
UpdateTeamLocation	See Section 2.16.4.11.3.	

Table 2.16-74: UpdateHaltLocation Information.

2.16.4.10.5 CloseHaltDialog

CloseHaltDialog is called to close the team halt dialog, *dialog*. The function call is CloseHaltDialog(dialog). Table 2.16-75 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Calls		
Function	**Where Described**	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomToWindow	See Section 2.22.1.47.4.	
ThrowDialog	See Section 2.22.1.11.3.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	
Called By		
Function	**Where Described**	
TemaHaltEvent	See Section 2.16.4.10.3.	

Table 2.16-75: CloseHaltDialog Information.

2.16.4.11 model.c

Development:SIMNET:MCC:Maint:model.c

model.c contains routines which model time-based processes involving maintenance teams, such as breakdowns.

2.16.4.11.1 InitTeamState

InitTeamState initializes the state of a team, *team*. The function call is InitTeamState(team). Table 2.16-76 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
Calls		
Function	**Where Described**	
Random	Standard Quickdraw function for Macintosh.	
DisableTeam	See Section 2.16.4.11.5.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.16.1.1.1.	
SetUp	See Section 2.16.4.2.1.	

Table 2.16-76: InitTeamState Information.

2.16.4.11.2 CheckTimers

CheckTimers looks for time-driven changes in status. This routine performs time-based processing once per minute. For each maintenance team, a check is made for the team having moved, arrived, become broken, or become fixed. A check is made for the team having finished hitching up a disabled vehicle, and having completed a repair. The function call is CheckTimers(). Table 2.16-77 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
now	unsigned long	Standard C type.
i	register short	Standard C type.
j	register short	Standard C type.
team	register pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
repair	register RepairHandle	Development:SIMNET:MCC:Maint:MaintMac.h
nextUpdateTime	unsigned long	Standard C type.
Calls		
Function	**Where Described**	
GetDateTime	Standard Operating System Utility function for Macintosh.	
EnableTeam	See Section 2.16.4.11.6.	
UpdateTeamLocation	See Section 2.16.4.11.3.	
DisableTeam	See Section 2.16.4.11.5.	
ReportDispatch	See Section 2.16.1.1.3.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	
NotifyChangeState	See Section 2.16.4.11.8.	
RepairCompleted	See Section 2.16.4.11.4.	

Table 2.16-77: CheckTimers Information.

2.16.4.11.3 UpdateTeamLocation

UpdateTeamLocation updates the status of a displacing team, *team*. If the team has arrived at its destination, any pop-up halt dialogs are discarded and the user and host are notified. If the team has not yet arrived at its destination, an interpolation of its current location is periodicatlly updated. The function call is UpdateTeamLocation(now, team). Table 2.16-78 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
now	unsigned long	Standard C type.
team	register pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pt	pointer to LongPt	Development:SIMNET:libmac:longpt.h
str	array of 100 char	Standard C type.

Calls	
Function	**Where Described**
DiscardPopUpDialog	See Section 2.16.4.11.9.
ReportHalt	See Section 2.16.1.1.4.
UpdateTeamTableRow	See Section 2.16.4.5.12.
NotifyStateChange	See Section 2.16.4.11.8.
InterpolatePoints	See Section 2.22.1.23.4.
PointToMapCoordinates	See Section 2.22.1.26.2.
UpdateHaltLocation	See Section 2.16.4.10.4.
Called By	
Function	**Where Described**
CheckTimers	See Section 2.16.4.11.2.

Table 2.16-78: UpdateTeamLocation Information.

2.16.4.11.4 RepairCompleted

RepairCompleted notes that a repair, *repair*, has been completed. The host and user are notified, and the repair is marked as having been completed. Any currently displayed repair cancellation dialog is closed. If a repair selection dialog is currently displayed for this team, repairs of the same class as that just completed should become available for selection. The function call is RepairCompleted(repair). Table 2.16-79 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
repair	register RepairHandle	Development:SIMNET:MCC:Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
str	array of 100 char	Standard C type.
Calls		
Function	**Where Described**	
ReportRepair	See Section 2.16.1.1.5.	
UpdateRepairTableRow	See Section 2.16.4.8.12.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	
DiscardPopUpDialog	See Section 2.16.4.11.9.	
UpdateStartDialog	See Section 2.16.4.7.9.	
CountCaution	See Section 2.22.1.4.4.	
DisposDialog	Standard Dialog Manager function for Macintosh.	
LastCaution	See Section 2.22.1.4.3.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
CheckTimers	See Section 2.16.4.11.2.	

Table 2.16-79: RepairCompleted Information.

2.16.4.11.5 DisableTeam

DisableTeam marks a team, *team*, as temporarily disabled. The routine ensures that no more than 20% of the maintenance teams are disabled. Any current dialog pertaining to this team is taken down. If the team was travelling, the host and user are notified that it has stopped. The subsequent repair of the team is scheduled. The function call is DisableTeam(team). Table 2.16-80 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
numberDisabled	int	Standard C type.
i	register short	Standard C type.
tp	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
Calls		
Function	**Where Described**	
DiscardPopUpDialog	See Section 2.16.4.11.9.	
ReportHalt	See Section 2.16.1.1.4.	
NotifyStateChange	See Section 2.16.4.11.8.	
UpdateTeamTableRow	See Section 2.16.4.5.12.	
ScheduleDisableEvent	See Section 2.16.4.11.7.	
Called By		
Function	**Where Described**	
InitTeamState	See Section 2.16.4.11.1.	
CheckTimers	See Section 2.16.4.11.2.	

Table 2.16-80: DisableTeam Information.

2.16.4.11.6 EnableTeam

EnableTeam reenables a team, *team*, that has been repaired. The user is notified that the team is no longer disabled, and the subsequent breakdown of the team is scheduled. The function call is EnableTeam(team). Table 2.16-81 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h

Calls	
Function	**Where Described**
NotifyStateChange	See Section 2.16.4.11.8.
UpdateTeamTableRow	See Section 2.16.4.5.12.
ScheduleDisableEvent	See Section 2.16.4.11.7.
Called By	
Function	**Where Described**
CheckTimers	See Section 2.16.4.11.2.

Table 2.16-81: EnableTeam Information.

2.16.4.11.7 ScheduleDisableEvent

ScheduleDisableEvent determines the length of time before a team next becomes broken or fixed. The function call is ScheduleDisableEvent(team, diceRange, numberDie). Table 2.16-82 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
diceRange	int	Standard C type.
numberDie	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dice	register short	Standard C type.
i	register short	Standard C type.
Calls		
Function	**Where Described**	
Random	Standard Quickdraw function for Macintosh.	
Called By		
Function	**Where Described**	
DisableTeam	See Section 2.16.4.11.5.	
EnableTeam	See Section 2.16.4.11.6.	

Table 2.16-82: ScheduleDisableEvent Information.

2.16.4.11.8 NotifyStateChange

NotifyStateChange notifies the user of a change in the state of a team, *team*. *state* is the current state of the team. The function call is NotifyStateChange(team, state). Table 2.16-83 describes the parameters used, and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
state	pointer to char	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cp	pointer to char	Standard C type.
str[100]	char	Standard C type.
Calls		
Function	**Where Described**	
CountCaution	See Section 2.22.1.4.4.	
DisposDialog	Standard Dialog Manager function for Macintosh.	
LastCaution	See Section 2.22.1.4.3.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.16.1.1.1.	
CheckTimers	See Section 2.16.4.11.2.	
UpdateTeamLocation	See Section 2.16.4.11.3.	
DisableTeam	See Section 2.16.4.11.5.	
EnableTeam	See Section 2.16.4.11.6.	

Table 2.16-83: NotifyStateChange Information.

2.16.4.11.9 DiscardPopUpDialog

DiscardPopUpDialog discards any pop-up dialog for commanding a team, *team*, or cancelling a repair. The function call is DiscardPopUpDialog(team). Table 2.16-84 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
team	register pointer to TeamStatus	Development:SIMNET:MCC:Maint:MaintMac.h
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.16.1.1.1.	
ProcessResponse	See Section 2.16.1.1.2.	
UpdateTemaLocation	See Section 2.16.4.11.3.	
RepairCompleted	See Section 2.16.4.11.4.	
DisableTeam	See Section 2.16.4.11.5.	

Table 2.16-84: DiscardPopUpDialog Information.

2.16.4.12 data.c
Development:SIMNET:MCC:Maint:data.c

data.c contains data definitions for the Maintenance application. Table 2.16-85 describes the variables used by data.c.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
application	pointer to char	Standard C type.
authors	pointer to char	Standard C type.
copyright	pointer to char	Standard C type.
simnetM1Repairs	array of RepairDescriptor	Development:SIMNET:MCC: Maint:MaintMac.h
simnetM1RepairSet	RepairSet	Development:SIMNET:MCC: Maint:MaintMac.h
simnetM2Repairs	array of RepairDescriptor	Development:SIMNET:MCC: Maint:MaintMac.h
simnetM2RepairSet	RepairSet	Development:SIMNET:MCC: Maint:MaintMac.h
genericRepairs	array of RepairDescriptor	Development:SIMNET:MCC: Maint:MaintMac.h
genericRepairSet	RepairSet	Development:SIMNET:MCC: Maint:MaintMac.h
armorTeamRepairSet	pointer to RepairSet	Development:SIMNET:MCC: Maint:MaintMac.h
armorTeamCapabilities	TeamCapabilities	Development:SIMNET:MCC: Maint:MaintMac.h
maintTeams	array of numberBnRepairVehicles TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
selectedTeam	short	Standard C type.
currentTeam	pointer to TeamStatus	Development:SIMNET:MCC: Maint:MaintMac.h
currentRepair	RepairHandle	Development:SIMNET:MCC: Maint:MaintMac.h
zoomBaseWindow	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
zoomBaseRect	Rect	Development:THINK C: Mac #includes:MacTypes.h
popUpDialogState	pointer to DialogState	Development:SIMNET:libmac: dialog.h
(closePopUpDialog)()	void function call	Standard C
battleScheme	char	Standard C type.
bnForceID	char	Standard C type.
companyForceID	array of maxNumberCompanies char	Standard C type.
startupComplete	char	Standard C type.
repairDepot	MapCoordinates	Development:SIMNET:libmac: map.h
repairClassName	pointer to array of char	Standard C type.

Table 2.16-85: data.c Variable Information.

2.16.5 Appearance of Maint User Interface

2.16.5.1 Maint Pictures
Development:SIMNET:MCC:Maint:Maint Pictures

Maint Pictures contains resources that determine the appearance of the Maint application's user interface.

2.17 The FSE (Fire Support Element) Console

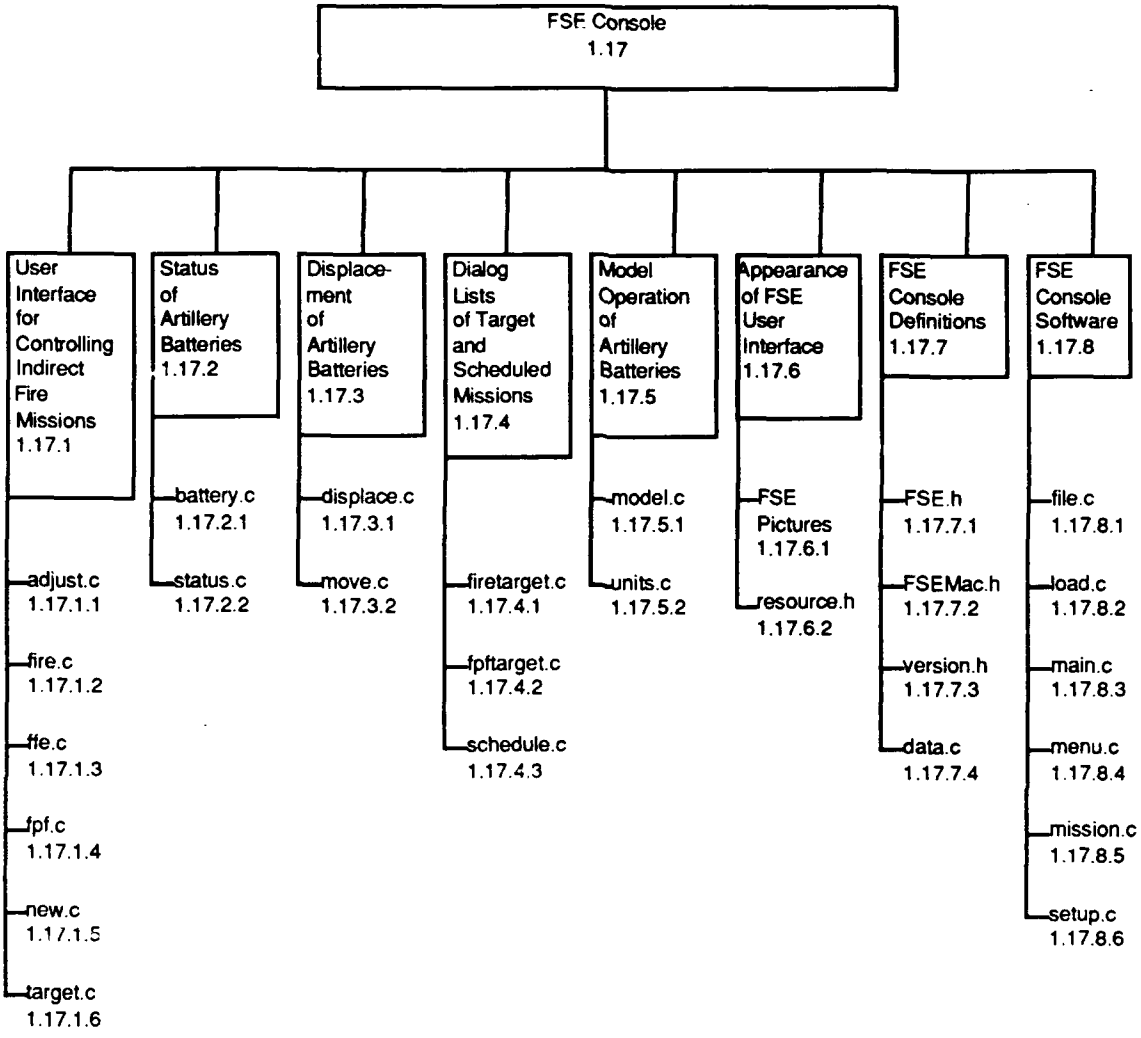


Figure 2.17-1: FSE (Fire Support Element) Console Structure.

2.17.1 User Interface for Controlling Indirect Fire Missions

2.17.1.1 adjust.c
Development:SIMNET:MCC:FSE:adjust.c

adjust.c implements adjust fire missions. Table 2.17-1 describes the variables used in adjust.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
adjustDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
adjustOTDirectionField	extern NumberFieldDefn	Development:SIMNET:libmac:dialog.h
rightButton	char	Standard C type.
dropButton	char	Standard C type.
leftRightDistance	int	Standard C type.
addDropDistance	int	Standard C type.
adjustOTDirectionField	NumberFieldDefn	Development:SIMNET:libmac:dialog.h
adjustRBFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
adjustDistanceFields	array of NumberFieldDefn	Development:SIMNET:libmac:dialog.h
adjustFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
adjustDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h
sinTable	array of int	Standard C type.

Table 2.17-1: adjust.c Variable Information.

2.17.1.1.1 ShowAdjust

ShowAdjust displays the adjust fire dialog. The function call is ShowAdjust(). Table 2.17-2 describes the functions called using this function.

Calls	
Function	**Where Described**
BatterySelectionViable	See Section 2.17.5.1.9.
ShowDialog	See Section 2.22.1.11.1.
ShowWindow	Standard Window Manager function for Macintosh.

Table 2.17-2: ShowAdjust Information.

2.17.1.1.2 AdjustEvent

AdjustEvent handles events in the Adjust Fire dialog. If Fire button is pressed, the routine first checks that the right combination of buttons and distances were used and that the mandatory fields were filled in. The routine then computes a new target location, starts the guns firing adjust rounds, and updates the display. If the FFEB button is pressed, the Adjust Fire dialog is taken down and the Fire For Effect dialog is popped. If the Cancel button is pressed, the Adjust Fire dialog is taken down. The function call is AdjustEvent(dialog, itemNo). Table 2.17-3 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pt	LongPt	Development:SIMNET:libmac:longpt.h
Calls		
Function	**Where Described**	
ShowCaution	See Section 2.22.1.4.1.	
CheckMandatoryFields	See Section 2.22.1.13.1.	
ComputeAdjustment	See Section 2.17.1.1.3.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
FireVolley	See Section 2.17.5.1.2.	
UpdateMissionDisplay	See Section 2.17.8.5.5.	
ThrowDialog	See Section 2.22.1.11.3.	
ShowFFEWithAdjust	See Section 2.17.1.3.2.	
SelIText	Standard Dialog Manager function for Macintosh.	

Table 2.17-3: AdjustEvent Information.

2.17.1.1.3 ComputeAdjustment

ComputeAdjustment applies an adjustment to a target location. The function call is ComputeAdjustment(pt, angle, lrButton, lsDistance, adButton, adDistance). Table 2.17-4 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
pt	pointer to LongPt	Development:SIMNET:libmac:longpt.h
angle	int	Standard C type.
lrButton	int	Standard C type.
lrDistance	int	Standard C type.
adButton	int	Standard C type.
adDistance	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sin	long	Standard C type.
cos	long	Standard C type.
Calls		
Function	**Where Described**	
ApproximateSin	See Section 2.17.1.1.4.	
ApproximateCos	See Section 2.17.1.1.5.	
Called By		
Function	**Where Described**	
AdjustEvent	See Section 2.17.1.1.2.	
NewMissionEvent	See Section 2.17.1.5.2.	
FFEEvent	See Section 2.17.1.3.3.	

Table 2.17-4: ComputeAdjustment Information.

2.17.1.1.4 ApproximateSin

ApproximateSin computes the approximate sin of *angle*. The sins for various angles are in *sinTable*. The function call is ApproximateSin(angle). Table 2.17-5 describes the parameter used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
angle	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
sin	register int	Standard C type.
a	register int	Standard C type.
Called By		
Function	**Where Described**	
ComputeAdjustment	See Section 2.17.1.1.3.	

Table 2.17-5: ApproximateSin Information.

2.17.1.1.5 ApproximateCos

ApproximateCos computes the approximate cos of *angle*. The function call is ApproximateCos(angle). Table 2.17-6 describes the parameter used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
angle	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
cos	register int	Standard C type.
a	register int	Standard C type.

Called By	
Function	**Where Described**
ComputeAdjustment	See Section 2.17.1.1.3.

Table 2.17-6: ShowAdjust Information.

2.17.1.2 fire.c

Development:SIMNET:MCC:FSE:fire.c

fire.c implements a dialog presenting the status of a indirect fire mission. Table 2.17-7 describes the variables used in fire.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
fireDialogDefn	extern DialogDefn	Development:SIMNET:libmac: dialog.h
fireDialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
fireRect	Rect	Development:THINK C: Mac #includes:MacTypes.h
buttonDefn[5][2]	matrix of struct buttonDefn	Development:SIMNET:MCC: FSE:fire.c
fireEndField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
fireObserverField	TextFieldDefn	Development:SIMNET:libmac: dialog.h
FireOTDirectionField	NumberFieldDefn	Development:SIMNET:libmac: dialog.h
fireAmmoFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
fireBatteryFields	array of CBFieldDefn	Development:SIMNET:libmac: dialog.h
fireFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
fireDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.17-7: fire.c Variable Information.

2.17.1.2.1 SetUpFireMission

SetUpFireMission initializes a dialog for displaying the fire mission status. The function call is SetUpFireMission(). Table 2.17-8 describes the functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.

Calls	
Function	**Where Described**
ShowDialog	See Section 2.22.1.11.1.
EstablishBatterySelection	See Section 2.17.5.2.1.
Called By	
Function	**Where Described**
main	See Section 2.17.8.3.1.

Table 2.17-8: SetUpFireMission Information.

2.17.1.2.2 GetFireDialogInfo

GetFireDialogInfo returns information about the fire dialog. The function call is GetFireDialogInfo(dialog, firstItemNo). Table 2.17-9 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogPtr	Development:THINK C: Mac #includes:DialogMgr.h
firstItemNo	pointer to int	Standard C type.
Called By		
Function	**Where Described**	
UpdateBatterySelection	See Section 2.17.5.2.2.	

Table 2.17-9: GetFireDialogInfo Information.

2.17.1.2.3 OpenFireMission

OpenFireMission opens an icon to display the status of the fire mission, *m*. The function call is OpenFireMission(m). Table 2.17-10 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
SelectWindow	Standard Window Manager function for Macintosh.	
ZoomMissionIcon	See Section 2.17.2.2.7.	
ShowFireMission	See Section 2.17.1.2.4.	
Called By		
Function	**Where Described**	
OpenMission	See Section 2.17.8.5.3.	

Table 2.17-10: OpenFireMission Information.

2.17.1.2.4 ShowFireMission

ShowFireMission makes the fire mission status dialog visible. The function call is ShowFireMission(m). Table 2.17-11 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
LoadFireBuffers	See Section 2.17.1.2.6.	
UpdateFireDialog	See Section 2.17.1.2.7.	
UpdateBatterySelection	See Section 2.17.5.2.2.	
SelectWindow	Standard Window Manager function for Macintosh.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
OpenFireMission	See Section 2.17.1.2.3.	
NewMissionEvent	See Section 2.17.1.5.2.	

Table 2.17-11: ShowFireMission Information.

2.17.1.2.5 FireEvent

FireEvent handles events in the Fire Mission dialog. If the End button is pressed, the mission is ended. If the OK button is pressed, the dialog is taken down. The function call is FireEvent(dialog, itemNo). Table 2.17-12 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
f	FireTargetHandle	Development:SIMNET:MCC: FSE: FSEMac.h
i	int	Standard C type.

Calls	
Function	**Where Described**
EndFireMission	See Section 2.17.1.2.10.
HideWindow	Standard Window Manager function for Macintosh.
ZoomMissionIcon	See Section 2.17.2.2.7.
NewFireTarget	See Section 2.17.4.1.1.
ShowFireTarget	See Section 2.17.4.1.11.
ToggleBattery	See Section 2.17.8.5.4.

Table 2.17-12: FireEvent Information.

2.17.1.2.6 LoadFireBuffers

LoadFireBuffers makes field definitions for the fire mission dialog point to locations of values for the appropriate mission. The function call is LoadFireBuffers(). Table 2.17-13 describes the internal variables used and function called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	pointer to register Mission	Development:SIMNET:MCC: FSE:FSEMac.h
i	register short	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
ShowFireMission	See Section 2.17.1.2.4.	

Table 2.17-13: LoadFireBuffers Information.

2.17.1.2.7 UpdateFireDialog

UpdateFireDialog updates the dialog displaying detailed status about a fire mission. The target location is filled in, the fuze selection is enabled or disabled, the status description is filled in, and the push buttons are labeled for the proper state. The function call is UpdateFireDialog(). Table 2.17-14 describes the functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
cp	register pointer to char	Standard C type.
i	int	Standard C type.
ctl	ControlHandle	Development:THINK C: Mac #includes:ControlMgr.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
buf	array of 80 char	Standard C type.

Calls	
Function	**Where Described**
SetText	See Section 2.22.1.11.8.
EnableControl	See Section 2.22.1.7.2.
DisableControl	See Section 2.22.1.7.1.
UpdateFireState	See Section 2.17.1.2.8.
GetDItem	Standard Dialog Manager function for Macintosh.
HideControl	Standard Control Manager function for Macintosh.
ShowControl	Standard Control Manager function for Macintosh.
SetCTitile	Standard Control Manager function for Macintosh.
UpdateDialog	See Section 2.22.1.11.2.
Called By	
Function	**Where Described**
ShowFireMission	See Section 2.17.1.2.4.
UpdateMissionDisplay	See Section 2.17.8.5.5.

Table 2.17-14: UpdateFireDialog Information.

2.17.1.2.8 UpdateFireState

UpdateFireState updates the display of a fire mission's state. The function call is UpdateFireState(). Table 2.17-15 describes the function called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
cp	register pointer to char	Standard C type.
buf	array of 80 char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
UpdateFireDialog	See Section 2.17.1.2.7.	
UpdateMissionState	See Section 2.17.8.5.6.	

Table 2.17-15: UpdateFireState Information.

2.17.1.2.9 UpdateOTDirection

UpdateOTDirection updates the display of the OT Direction in the mission status dialog. This function is called by the libmac dialog code as a field validation routine. It always returns 1. The function call is UpdateOTDirection(dialog, fp, str). Table 2.17-16 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
fp	pointer to TypeInFieldDefn	Development:SIMNET:libmac: dialog.h
str	pointer to char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
1	int	Always returned.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	

Table 2.17-16: UpdateOTDirection Information.

2.17.1.2.10 EndFireMission

EndFireMission ends a fire mission, *m*. A dialog is brought down, displaying the mission, if any. The batteries and the mission descriptor aregdgf freed. The function call is EndFireMission(m). Table 2.17-17 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
ApplyToBatteries	See Section 2.17.5.1.5.	
EndMission	See Section 2.17.8.5.2.	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomMissionIcon	See Section 2.17.2.2.7.	
Called By		
Function	**Where Described**	
FireEvent	See Section 2.17.1.2.5.	
FireVolley	See Section 2.17.5.1.2.	

Table 2.17-17: EndFireMission Information.

2.17.1.3 ffe.c
Development:SIMNET:MCC:FSE:ffe.c

ffe.c implements dialogs controlling fire for effect missions. Table 2.17-18 describes the variables used in ffe.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
ffeDialogDefn	externDialogDefn	Development:SIMNET:libmac:dialog.h
ffeOTDirectionField	extern NumberFieldDefn	Development:SIMNET:libmac:dialog.h
ffeRoundsField	extern NumberFieldDefn	Development:SIMNET:libmac:dialog.h
rightButton	char	Standard C type.
dropButton	char	Standard C type.
leftRightDistance	int	Standard C type.
addDropDistance	int	Standard C type.
controlBuffer	char	Standard C type.
ffeOTDirectionField	NumberFieldDefn	Development:SIMNET:libmac:dialog.h
ffeRBFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
ffeDistanceFields	array of NumberFieldDefn	Development:SIMNET:libmac:dialog.h
ffeRoundsField	NumberFieldDefn	Development:SIMNEt:libmac:dialog.h
ffeControlFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
ffeFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
ffeDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.17-18: ffe.c Variable Information.

2.17.1.3.1 ShowFFE

ShowFFE displays the Fire For Effect dialog. The function call is ShowFFE(). Table 2.17-19 describes the functions called using this function.

Calls	
Function	**Where Described**
BatterySelectionViable	See Section 2.17.5.1.9.
ShowFFEWithAdjust	See Section 2.17.1.3.2.

Table 2.17-19: ShowFFE Information.

2.17.1.3.2 ShowFFEWithAdjust

ShowFFEWithAdjust shows the Fire For Effect dialog with partially completed adjustment. The function call is ShowFFEWithAdjust(lrButton, lrDistance, adButton, adDistance). Table 2.17-20 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
lrButton	int	Standard C type.
lrDistance	int	Standard C type.
adButton	int	Standard C type.
adDistance	int	Standard C type.
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
AdjustEvent	See Section 2.17.1.1.2.	
ShowFFE	See Section 2.17.1.3.1.	

Table 2.17-20: ShowFFEWithAdjust Information.

2.17.1.3.3 FFEEvent

FFEEvent handles events in the Fire For Effect dialog. If the Fire button is pressed, the routine checks for validity of the data, updates the coordinates (in case the target changed), computes the new target loaction, and starts the guns firing volleys. If the Cancel button is pressed, the dialog is taken down. The function call is FFEEvent(dialog, itemNo). Table 2.17-21 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pt	LongPt	Development:SIMNET:libmac: longpt.h
Calls		
Function	**Where Described**	
ShowCaution	See Section 2.22.1.4.1.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
CheckMandatoryFields	See Section 2.22.1.13.1.	
FireVolley	See Section 2.17.5.1.2.	
ComputeAdjustment	See Section 2.17.1.1.3.	
UpdateMissionDisplay	See Section 2.17.8.5.5.	
ThrowDialog	See Section 2.22.1.11.3.	
SellText	Standard Dialog Manager function for Macintosh.	

Table 2.17-21: FFEEvent Information.

2.17.1.3.4 CommenceFFE

CommenceFFE starts a Fire For Effect waiting for an "at my command". The function call is CommenceFFE(). Table 2.17-22 describes the functions called using this function.

Calls	
Function	**Where Described**
FireVolley	See Section 2.17.5.1.2.
UpdateMissionDisplay	See Section 2.17.8.5.5.

Table 2.17-22: CommenceFFE Information.

2.17.1.3.5 CancelAtCommand

CancelAtCommand cancels a Fire For Effect, which was waiting for an "at my command". The function call is CancelAtCommand(). Table 2.17-23 describes the function called using this function.

Calls	
Function	**Where Described**
UpdateMissionDisplay	See Section 2.17.8.5.5.

Table 2.17-23: CancelAtCommand Information.

2.17.1.3.6 CheckFiring

CheckFiring cancels a Fire For Effect in progress. If the mission was scheduled in advance, it is disassociated from the fire mission schedule and placed under manual control. The function call is CheckFiring(). Table 2.17-24 describes the functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
ApplyToBatteries	See Section 2.17.5.1.5.	
UpdateMissionDisplay	See Section 2.17.8.5.5.	
SetSchedMission	See Section 2.17.4.3.16.	

Table 2.17-24: CheckFiring Information.

2.17.1.4 fpf.c

Development:SIMNET:MCC:FSE:fpf.c

fpf.c implements dialogs controlling final protective fire missions. Table 2.17-25 describes the variables used in fpf.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
fpfDialogDefn	extern DialogDefn	Development:SIMNET:libmac: dialog.h
fpfdialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
fpfRect	Rect	Development:THINK C: Mac #includes:MacTypes.h
fpfEndField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
fpfObserverField	TextFieldDefn	Development:SIMNET:libmac: dialog.h
fpfNumberField	NumberFieldDefn	Development:SIMNET:libmac: dialog.h
fpfBatteryFields	array of CBFieldDefn	Development:SIMNET:libmac: dialog.h
fpfFieldList	array of pointer to FieldDefn	Development:SIMNET:libmac: dialog.h
fpfDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.17-25: fpf.c Variable Information.

2.17.1.4.1 SetUpFPFMission

SetUpFPFMission initializes a dialog for displaying FPF mission status. The function call is SetUpFPFMission(). Table 2.17-26 describes the functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
EstablishBatterySelection	See Section 2.17.5.2.1.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-26: SetUpFPFMission Information.

2.17.1.4.2 GetFPFDialogInfo

GetFPFDialogInfo returns information about the FPF dialog. The function call is GetFPFDialogInfo(dialog, firstItemNo). Table 2.17-27 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogPtr	Development:THINK C: Mac #includes:DialogMgr.h
firstItemNo	pointer to int	Standard C type.
Called By		
Function	**Where Described**	
UpdateBatterySelection	See Section 2.17.5.2.2.	

Table 2.17-27: GetFPFDialogInfo Information.

2.17.1.4.3 ShowNewFPFMission

ShowNewFPFMission puts up a dialog for a new FPF mission. The function call is ShowNewFPFMission(). Table 2.17-28 describes the functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
NewMission	See Section 2.17.8.5.1.	
RedrawMissionIcon	See Section 2.17.2.2.5.	
ShowFPFMission	See Section 2.17.1.4.5.	
Called By		
Function	**Where Described**	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-28: ShowNewFPFMission Information.

2.17.1.4.4 OpenFPFMission

OpenFPFMission opens an icon to display the status of an FPF mission, *m*. The function call is OpenFPFMission(m). Table 2.17-29 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h

Calls	
Function	**Where Described**
SelectWindow	Standard Window Manager function for Macintosh.
ZoomMissionIcon	See Section 2.17.2.2.7.
ShowFPFMission	See Section 2.17.1.4.5.
Called By	
Function	**Where Described**
OpenMission	See Section 2.17.8.5.3.

Table 2.17-29: OpenFPFMission Information.

2.17.1.4.5 ShowFPFMission

ShowFPFMission makes an FPF mission status dialog visible. The function call is ShowFPFMission(m). Table 2.17-30 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
LoadFPFBuffers	See Section 2.17.1.4.8.	
UpdateFPFDialog	See Section 2.17.1.4.9.	
UpdateBatterySelection	See Section 2.17.5.2.2.	
SelectWindow	Standard Window Manager function for Macintosh.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
ShowNewFPFMission	See Section 2.17.1.4.3.	
OpenFPFMission	See Section 2.17.1.4.4.	

Table 2.17-30: ShowFPFMission Information.

2.17.1.4.6 FPFEvent

FPFEvent handles events in the FPF dialog. The function call is FPFEvent(dialog, itemNo). Table 2.17-31 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Sevelopment:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
Calls		
Function	**Where Described**	
ApplyToBatteries	See Section 2.17.5.1.5.	
EndMission	See Section 2.17.8.5.2.	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomMissionIcon	See Section 2.17.2.2.7.	
CheckMandatoryFields	See Section 2.22.1.13.1.	
BatterySelectionViable	See Section 2.17.5.1.9.	
FireVolley	See Section 2.17.5.1.2.	
UpdateMissionDisplay	See Section 2.17.8.5.5.	
ToggleBattery	See Section 2.17.8.5.4.	

Table 2.17-31: FPFEvent Information.

2.17.1.4.7 ValidFPFTarget

ValidFPFTarget determines whether an FPF target number is valid. The function call is ValidFPFTarget(dialog, fp, str). Table 2.17-32 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
fp	pointer to TypeInFieldDefn	Development:SIMNET:libmac: dialog.h
str	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
fpfTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
f	register FPFTargetHandle	Development:SIMNET:MCC: FSE: FSEMac.h
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
Return Values		
Return Value	**Type**	**Meaning**
0	int	FPF target number not valid.
1	int	FPFtarget number valid.
Calls		
Function	**Where Described**	
LookupScrollTableEntry	See Section 2.22.1.24.1.	
ShowCaution	See Section 2.22.1.4.1.	
RedrawMissionIcon	See Section 2.17.2.2.5.	

Table 2.17-32: ValidFPFTarget Information.

2.17.1.4.8 LoadFPFBuffers

LoadFPFBuffers makes field definitions for FPF mission dialog point to locations of values for the appropriate mission. The function call is LoadFPFBuffers(). Table 2.17-33 describes the internal variables used by this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
i	register short	Standard C type.
Called By		
Function	**Where Described**	
ShowFPFMission	See Section 2.17.1.4.5.	

Table 2.17-33: LoadFPFBuffers Information.

2.17.1.4.9 UpdateFPFDialog

UpdateFPFDialog updates the dialog displaying detailed status about an FPF mission. The function call is UpdateFPFDialog(). Table 2.17-34 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
i	int	Standard C type.
theItem	Handle	Development:THINK C:Mac #includes:MacTypes.h
box	Rect	Development:THINK C:Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
GetDItem	Standard Dialog Manager function for Macintosh.	
SetDItem	Standard Dialog Manager function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
InsetRect	Standard Quickdraw function for Macintosh.	
EraseRect	Standard Quickdraw function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
SelIText	Standard Dialog Manager function for Macintosh.	
SetCTitle	Standard Control Manager function for Macintosh.	
UpdateDialog	See Section 2.22.1.11.2.	
Called By		
Function	**Where Described**	
ShowFPFMission	See Section 2.17.1.4.5.	
UpdateMissionDisplay	See Section 2.17.8.5.5.	

Table 2.17-34: UpdateFPFDialog Information.

2.17.1.5 new.c

Development:SIMNET:MCC:FSE:new.c

new.c implements a dialog that prompts for information describing a new fire mission. Table 2.17-35 describes the variables used in new.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
newMissionDialogDefn	extern DialogDefn	Development:SIMNET:libmac: dialog.h
newMissionLocationFields	extern array of TextFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionDirectionFields	extern array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionDistanceFields	extern array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
observerBuffer	array of observerLength char	Standard C type.
tgtBuffer	char	Standard C type.
locnBuffer	array of 3 TargetLocationBuffer	Development:SIMNET:MCC: FSE:FSEMac.h
directionBuffer	array of 2 int	Standard C type.
otDistanceBuffer	int	Standard C type.
lrDistanceBuffer	int	Standard C type.
adDistanceBuffer	int	Standard C type.
lrButtonBuffer	char	Standard C type.
adButtonBuffer	char	Standard C type.
descriptionBuffer	array of descriptionLength char	Standard C type.
newMissionCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionObserverField	TextFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionTgtButtonsFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionLocationFields	array of TextFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionDirectionFields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionDistanceFields	array of NumberFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionShiftBtnFields	array of RBFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionDescriptionField	TextFieldDefn	Development:SIMNET:libmac: dialog.h
newMissionFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
newMissionDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.17-35: new.c Variable Information.

2.17.1.5.1 ShowNewFireMission

ShowNewFireMission puts up a dialog to take target information for a new fire mission. The function call is ShowNewFireMission(). Table 2.17-36 describes the functions called using this function.

Calls	
Function	**Where Described**
NewMission	See Section 2.17.8.5.1.
RedrawMissionIcon	See Section 2.17.2.2.5.
ShowDialog	See Section 2.22.1.11.1.
ShowWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-36: ShowNewFireMission Information.

2.17.1.5.2 NewMissionEvent

NewMissionEvent handles events in the New Mission dialog. The routine checks for the validity of the data and performs the appropriate action associated with the button pressed. The function call is NewMissionEvent(dialog, itemNo). Table 2.17-37 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
locn	TargetLocation	Development:SIMNET:MCC: FSE: FSEMac.h
pt	LongPt	Development:SIMNET:libmac: longpt.h
otd	int	Standard C type.

Calls	
Function	**Where Described**
ShowCaution	See Section 2.22.1.4.1.
CheckMandatoryFields	See Section 2.22.1.13.1.
ComputeAdjustment	See Section 2.17.1.1.3.
PointToMapCoordinates	See Section 2.22.1.26.2.
RedrawMissionIcon	See Section 2.17.2.2.5.
ThrowDialog	See Section 2.22.1.11.3.
ShowFireMission	See Section 2.17.1.2.4.
EndMission	See Section 2.17.8.5.2.
SellText	Standard Dialog Manager function for Macintosh.

Table 2.17-37: NewMissionEvent Information.

2.17.1.6 target.c

Development:SIMNET:MCC:FSE:target.c

target.c contains a routine that parses a string specifying a target location, either as by UTM coordinates or fire target number.

2.17.1.6.1 ValidTgtLocation

ValidTgtLocation is the target validation routine for the dialogs. The function call is ValidTgtLocation(dialog, fp, str). Table 2.17-38 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
fp	pointer to TextFieldDefn	Development:SIMNET:libmac: dialog.h
str	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
buf	register pointer to TargetLocationBuffer	Development:SIMNET:MCC: FSE: FSEMac.h
i	register short	Standard C type.
ch	char	Standard C type.
msg	array of 80 char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
1	int	Succesful
0	int	Unsuccessful

Calls	
Function	**Where Described**
LookupFireTarget	See Section 2.17.4.1.3.
StringToMapCoordinates	See Section 2.22.1.26.1.
ShowCaution	See Section 2.22.1.4.1.

Table 2.17-38: ValidTgtLocation Information.

2.17.2 Status of Artillery Batteries

2.17.2.1 battery.c

Development:SIMNET:MCC:FSE:battery.c

battery.c implements a dialog displaying the status of an artillery battery. Table 2.17-39 describes the variables used in battery.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
currentBattery	pointer to Battery	Development:SIMNET:MCC: FSE:FSEMac.h
batteryDialog	DialogPtr	Development:THINK C: Mac #includes:DialogMgr.h
batteryRect	Rect	Development:THINK C: Mac #includes:MacTypes.h

Table 2.17-39: battery.c Variable Information.

2.17.2.1.1 InstallDrawRoutine

InstallDrawRoutine installs the custom draw routines used by the battery status dialog. The function call is InstallDrawRoutine(itemNo, fnc). Table 2.17-40 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
itemNo	int	Standard C type.
(fnc)()	pointer to void function	Standard
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
GetDItem	Standard Dialog Manager function for Macintosh.	
SetDItem	Standard Dialog Manager function for Macintosh.	

Called By	
Function	**Where Described**
SetUpBattery	See Section 2.17.2.1.2.

Table 2.17-40: InstallDrawRoutine Information.

2.17.2.1.2 SetUpBattery

SetUpBattery initializes a dialog for displaying the battery status. The function call is SetUpBattery(). Table 2.17-41 describes the functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
itemNos	array of char	Standard C type.
Calls		
Function	**Where Described**	
GetNewDialog	Standard Dialog Manager function for Macintosh.	
OutlineItem	See Section 2.22.1.27.2.	
InstallDrawRoutine	See Section 2.17.2.1.1.	
SetWRefCon	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-41: SetUpBattery Information.

2.17.2.1.3 ShowBattery

ShowBattery shows the dialog displaying the status of a gun battery, *b*. The function call is ShowBattery(b). Table 2.17-42 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	register pointer to Battery	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
SelectWindow	Standard Window Manager function for Macintosh.	
ZoomBatteryIcon	See Section 2.17.2.2.8.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
StatusEventHandler	See Section 2.17.2.2.4.	

Table 2.17-42: ShowBattery Information.

2.17.2.1.4 BatteryEventHandler

BatteryEventHandler handles an event in the battery status dialog. The function call is BatteryEventHandler(window, theEvent). Table 2.17-43 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
dp	DialogPtr	Development:THINK C: Mac #includes:DialogMgr.h
i	int	Standard C type.
Calls		
Function	**Where Described**	
DialogSelect	Standard Dialog Manager function for Macintosh.	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomBatteryIcon	See Section 2.17.2.2.8.	

Table 2.17-43: BatteryEventHandler Information.

2.17.2.1.5 DrawBatteryText

DrawBatteryText draws a text field in the battery dialog. The function call is DrawBatteryText(theWindow, itemNo). Table 2.17-44 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theWindow	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
itemNo	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
b	register pointer to Battery	Development:SIMNET:MCC: FSE:FSEMac.h
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
cp	pointer to char	Standard C type.
str	array of 100 char	Standard C type.
Calls		
Function	**Where Described**	
GetDItem	Standard Dialog Manager function for Macintosh.	
MortarBattery	Macro defined in FSEMac.h. See Section 2.17.7.2.	
SetPort	Standard Quickdraw function for Macintosh.	
TextBox	Standard TextEdit function for Macintosh.	

Table 2.17-44: DrawBatteryText Information.

2.17.2.1.6 DrawBatteryRect

DrawBatteryRect draws a rectangle in the battery dialog. The function call is DrawBatteryRect(theWindow, itemNo). Table 2.17-45 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theWindow	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
GetDItem	Standard Dialog Manager function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
PenNormal	Standard Quickdraw function for Macintosh.	
FrameRect	Standard Quickdraw function for Macintosh.	

Table 2.17-45: DrawBatteryRect Information.

2.17.2.1.7 UpdateBatteryDisplay

UpdateBatteryDisplay updates the screen for a battery, specified by *battery* and *half*, whose status may have changed. The function call is UpdateBatteryDisplay(battery, half). Table 2.17-46 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
r	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
RedrawBatteryIcon	See Section 2.17.2.2.6.	
SetPort	Standard Quickdraw function for Macintosh.	
GetDItem	Standard Dialog Manager function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
LocalUnitDispatched	See Section 2.17.3.2.1.	
LocalUnitArrived	See Section 2.17.3.2.2.	
RemoteUnitDispatched	See Section 2.17.3.2.3.	
RemoteUnitArrived	See Section 2.17.3.2.4.	
UpdateUnitLocation	See Section 2.17.3.2.6.	
CheckTimers	See Section 2.17.5.1.1.	
StartMissionBattery	See Section 2.17.5.1.6.	
EndMissionBattery	See Section 2.17.5.1.7.	
ProcessRequest	See Section 2.17.8.3.6.	

Table 2.17-46: UpdateBatteryDisplay Information.

2.17.2.1.8 UpdateAmmoDisplay

UpdateAmmoDisplay updates the display of a battery's ammo supply, if visible. The function call is UpdateAmmoDisplay(b). Table 2.17-47 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
theItem	Handle	Development:THINK C: Mac #includes:MacTypes.h
r	Rect	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
SetPort	Standard Quickdraw function for Macintosh.	
GetDItem	Standard Dialog Manager function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
FireVolley	See Section 2.17.5.1.2.	
ProcessRequest	See Section 2.17.8.3.6.	

Table 2.17-47: UpdateAmmoDisplay Information.

2.17.2.2 status.c

Development:SIMNET:MCC:FSE:status.c

status.c implements a window displaying the status of all artillery batteries and active fire missions. Table 2.17-48 describes the variables used in status.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
statusWindow	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
headingBaseline	short	Standard C type.
columnCentre	array of numberOfColumns short	Standard C type.
textHeight	short	Standard C type.
rowBase	array of numberOfRows struct RowBase	Development:SIMNET:MCC: FSE:status.c
missionPix	array of maxNumberMissions struct MissionPix	Development:SIMNET:MCC: FSE:status.c
batteryPix	array of maxNumberBatteries struct BatteryPix	Development:SIMNET:MCC: FSE:status.c
missionIcons	array of 3 Handle	Development:THINK C: Mac #includes:MacTypes.h

Table 2.17-48: status.c Variable Information.

2.17.2.2.1 SetUpStatusWindow

SetUpStatusWindow creates and displays the status window. Each fire unit status icon, each row of mission status icons, and each mission status is located. The icons are loaded from the resource files. The function call is SetUpStatusWindow(). Table 2.17-49 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
m	pointer to struct MissionPix	Development:SIMNET:MCC: FSE:status.c
b	pointer to struct BatteryPix	Development:SIMNET:MCC: FSE:status.c
r	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
finfo	FontInfo	Development:THINK C: Mac #includes:Quickdraw.h
columnWidth	short	Standard C type.
columnHalfWidth	short	Standard C type.
centre	short	Standard C type.
leftEdge	short	Standard C type.
rightEdge	short	Standard C type.
iconTop	short	Standard C type.
iconBottom	short	Standard C type.
nameTop	short	Standard C type.
nameBottom	short	Standard C type.
statusTop	short	Standard C type.
statusBottom	short	Standard C type.
targetTop	short	Standard C type.
targetBottom	short	Standard C type.
rowHeight	short	Standard C type.
iconResourceID	array of int	Standard C type.
Calls		
Function	**Where Described**	
GetNewWindow	Standard Window Manager function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
SetWRefCon	Standard Window Manager function for Macintosh.	
TextFont	Standard Quickdraw function for Macintosh.	
TextSize	Standard Quickdraw function for Macintosh.	
GetFontInfo	Standard Quickdraw function for Macintosh.	
SetRect	Standard Quickdraw function for Macintosh.	
LocalToGlobal	Standard Quickdraw function for Macintosh.	
GetIcon	Standard Toolbox Utility function for Macintosh.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-49: SetUpStatusWindow Information.

2.17.2.2.2 ShowStatusWindow

ShowStatusWindow makes the status window visible. The function call is ShowStatusWindow(). Table 2.17-50 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
SelectWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
main	See Section 2.17.8.3.1.
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-50: ShowStatusWindow Information.

2.17.2.2.3 DrawStatusWindow

DrawStatusWindow draws the contents of the status window. Headings, battery status, and mission status are drawn. The function call is DrawStatusWindow(). Table 2.17-51 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
mp	register pointer to struct MissionPix	Development:SIMNET:MCC: FSE:status.c
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
b	pointer to struct BatteryPix	Development:SIMNET:MCC: FSE:status.c
r	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
j	short	Standard C type.
cp	pointer to char	Standard C type.
str	array of 10 char	Standard C type.
stateString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
SetPort	Standard Quickdraw function for Macintosh.	
TextSize	Standard Quickdraw function for Macintosh.	
TextFont	Standard Quickdraw function for Macintosh.	
MoveTo	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	
PenSize	Standard Quickdraw function for Macintosh.	
Line	Standard Quickdraw function for Macintosh.	
EraseRect	Standard Quickdraw function for Macintosh.	
PlotIcon	Standard Toolbox Utility function for Macintosh.	
TextBox	Standard TextEdit function for Macintosh.	

Called By	
Function	**Where Described**
StatusEventHandler	See Section 2.17.2.2.4.

Table 2.17-51: DrawStatusWindow Information.

2.17.2.2.4 StatusEventHandler

StatusEventHandler handles an event in the status window. The function call is StatusEventHandler(window, theEvent). Table 2.17-52 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
window	WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
theEvent	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
i	register short	Standard C type.
pt	Point	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
GlobalToLocal	Standard Quickdraw function for Macintosh.	
PtInRect	Standard Quickdraw function for Macintosh.	
OpenMission	See Section 2.17.8.5.3.	
ShowBattery	See Section 2.17.2.1.3.	
BeginUpdate	Standard Window Manager function for Macintosh.	
DrawStatusWindow	See Section 2.17.2.2.3.	
EndUpdate	Standard Window Manager function for Macintosh.	

Table 2.17-52: StatusEventHandler Information.

2.17.2.2.5 RedrawMissionIcon

RedrawMissionIcon forces an update of the mission status in the status window. The function call is RedrawMissionIcon(m). Table 2.17-53 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
ShowNewFPFMission	See Section 2.17.1.4.3.	
ValidFPFTarget	See Section 2.17.1.4.7.	
ShowNewFireMission	See Section 2.17.1.5.1.	
NewMissionEvent	See Section 2.17.1.5.2.	
UpdateObserver	See Section 2.17.8.5.7.	
UpdateMissionState	See Section 2.17.8.5.6.	
UpdateMissionDisplay	See Section 2.17.8.5.5.	
EndMission	See Section 2.17.8.5.2.	
UnhookFireTarget	See Section 2.17.4.1.15.	

Table 2.17-53: RedrawMissionIcon Information.

2.17.2.2.6 RedrawBatteryIcon

RedrawBatteryIcon forces an update of the battery status in the status window. The function call is RedrawBatteryIcon(b, half). Table 2.17-54 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.
half	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
savePort	GrafPtr	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
GetPort	Standard Quickdraw function for Macintosh.	
SetPort	Standard Quickdraw function for Macintosh.	
InvalRect	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	

Table 2.17-54: RedrawBatteryIcon Information.

2.17.2.2.7 ZoomMissionIcon

ZoomMissionIcon zooms to or from a mission icon. The function call is ZoomMissionIcon(m, bigRect, zoomUp). Table 2.17-55 describes the parameters used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
bigRect	pointer to Rect	Development:THINK C:Mac #includes:MacTypes.h
zoomUp	int	Standard C type.
Calls		
Function	**Where Described**	
ZoomRect	See Section 2.22.1.47.2.	
Called By		
Function	**Where Described**	
OpenFireMission	See Section 2.17.1.2.3.	
FireEvent	See Section 2.17.1.2.5.	
EndFireMission	See Section 2.17.1.2.10.	
OpenFPFMission	See Section 2.17.1.4.4.	
FPFEvent	See Section 2.17.1.4.6.	

Table 2.17-55: ZoomMissionIcon Information.

2.17.2.2.8 ZoomBatteryIcon

ZoomBatteryIcon zooms to or from a battery icon. The function call is ZoomBatteryIcon(b, bigRect, zoomUp). Table 2.17-56 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.
bigRect	pointer to Rect	Development:THINK C:Mac #includes:MacTypes.h
zoomUp	int	Standard C type.
Calls		
Function	**Where Described**	
ZoomRect	See Section 2.22.1.47.2.	
Called By		
Function	**Where Described**	
ShowBattery	See Section 2.17.2.1.3.	
BatteryEventHandler	See Section 2.17.2.1.4.	

Table 2.17-56: ZoomBatteryIcon Information.

2.17.3 Displacement of Artillery Batteries

2.17.3.1 displace.c

Development:SIMNET:MCC:FSE:displace.c

displace.c implements dialogs controlling mortar displacement. Table 2.17-57 describes the variables used in displace.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
displacedialogDefn	extern DialogDefn	Development:SIMNET:libmac: dialog.h
abortDialogDefn	extern DialogDefn	Development:SIMNET:libmac: dialog.h
displaceDisplayed	char	Standard C type.
displaceBattery	int	Standard C type.
displaceHalf	int	Standard C type.
destLocnBuffer	MapCoordinates	Development:SIMNET:libmac: map.h
destAzimuthBuffer	int	Standard C type.
spTimeBuffer	DateTimeGroup	Development:SIMNET:libmac: dtg.h
rpTimeBuffer	DateTimeGroup	Development:SIMNET:libmac: dtg.h
displaceCancelField	PushButtonFieldDefn	Development:SIMNET:libmac: dialog.h
displaceDestLocnField	CoordinatesFieldDefn	Development:SIMNET:libmac: dialog.h
displaceDestAzimuthField	NumberFieldDefn	Development:SIMNET:libmac: dialog.h
displaceSPTimeField	DTGFieldDefn	Development:SIMNET:libmac: dialog.h
displaceFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
displaceDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h
abortFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac: dialog.h
abortDialogDefn	DialogDefn	Development:SIMNET:libmac: dialog.h

Table 2.17-57: displace.c Variable Information.

2.17.3.1.1 ShowDisplace

ShowDisplace puts up the dialog which allows entry or cancelling of a mortar displacement order. The function call is ShowDisplace(battery, half). Table 2.17-58 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
MainEventLoop	See Section 2.17.8.3.3.	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-58: ShowDisplace Information.

2.17.3.1.2 ThrowDisplace

ThrowDisplace discards any mortar displacement dialog which is visible. The function call is ThrowDisplace(battery, half). Table 2.17-59 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.
Calls		
Function	**Where Described**	
ThrowDialog	See Section 2.22.1.11.3.	
Called By		
Function	**Where Described**	
AbortEvent	See Section 2.17.3.1.8.	
DisplaceEvent	See Section 2.17.3.1.5.	
LocalUnitDispatched	See Section 2.17.3.2.1.	
LocalUnitArrived	See Section 2.17.3.2.2.	
RemoteUnitDispatched	See Section 2.17.3.2.3.	
RemoteUnitArrived	See Section 2.17.3.2.4.	
ProcessRequest	See Section 2.17.8.3.6.	

Table 2.17-59: ThrowDisplace Information.

2.17.3.1.3 DisplaceFetch

DisplaceFetch reads the current data structures for the Schedule Displacement dialog into the dialog for display to the user. The function call is DisplaceFetch(dialog). Table 2.17-60 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
UpdateReleasePoint	See Section 2.17.3.1.4.	
Called By		
Function	**Where Described**	
UndoDisplace	See Section 2.17.3.1.6.	

Table 2.17-60: DisplaceFetch Information.

2.17.3.1.4 UpdateReleasePoint

UpdateReleasePoint updates the computed release point. The function call is UpdateReleasePoint(). Table 2.17-61 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of char	Standard C type.
Calls		
Function	**Where Described**	
DTGElapsed	See Section 2.22.1.15.3.	
DistBetween2Pts	See Section 2.22.1.23.2.	
DTGToString	See Section 2.22.1.15.1.	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
DisplaceEvent	See Section 2.17.3.1.5.	
DisplaceFetch	See Section 2.17.3.1.3.	

Table 2.17-61: UpdateReleasePoint Information.

2.17.3.1.5 DisplaceEvent

DisplaceEvent handles events in the Displace dialog. The function call is DisplaceEvent(dialog, itemNo). Table 2.17-62 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
UpdateReleasePoint	SeeSection 2.17.3.1.4.	
CheckMandatoryFields	See Section 2.22.1.13.1.	
ThrowDisplace	See Section 2.17.3.1.2.	

Table 2.17-62: DisplaceEvent Information.

2.17.3.1.6 UndoDisplace

UndoDisplace undoes any changes in the displace dialog. The function call is UndoDisplace(). Table 2.17-63 describes the parameters used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
DisplaceFetch	See Section 2.17.3.1.3.	
UpdateDialog	See Section 2.22.1.11.2.	
Called By		
Function	**Where Described**	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-63: UndoDisplace Information.

2.17.3.1.7 AbortFetch

AbortFetch reads current values into the Abort Displacement dialog. The function call is AbortFetch(dialog). Table 2.17-64 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
DTGToString	See Section 2.22.1.15.1.	
RedrawUnitLocation	See Section 2.17.3.1.9.	

Table 2.17-64 : AbortFetch Information.

2.17.3.1.8 AbortEvent

AbortEvent handles events in the Abort Displacement dialog. If the Do button is pressed, the immediate arrival of the fire unit is forced. If the Dont button is pressed, the dialog is taken down. The function call is AbortEvent(dialog, itemNo). Table 2.17-65 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
now	unsigned long	Standard C type.
Calls		
Function	**Where Described**	
GetDateTime	Standard Operating System Utility function for Macintosh.	
ThrowDisplace	See Section 2.17.3.1.2.	

Table 2.17-65: AbortEvent Information.

2.17.3.1.9 RedrawUnitLocation

RedrawUnitLocation redraws the location of a fire unit, specified by *battery* and *half*, if it is currently displayed in a displacement dialog. The function call is RedrawUnitLocation(battery, half). Table 2.17-66 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
SetText	See Section 2.22.1.11.8.	
Called By		
Function	**Where Described**	
AbortFetch	See Section 2.17.3.1.7.	
UpdateUnitLocation	See Section 2.17.3.2.6.	

Table 2.17-66: RedrawUnitLocation Information.

2.17.3.2 move.c

Development:SIMNET:MCC:FSE:move.c

move.c contains routines that model the displacement of artillery batteries.

2.17.3.2.1 LocalUnitDispatched

LocalUnitDispatched is called when a fire unit, specified by *battery* and *half*, is leaving on displacement due to an order entered at this console. The function call is LocalUnitDispatched(battery, half). Table 2.17-67 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
now	unsigned long	Standard C type.
delay	unsigned long	Standard C type.
ab	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
req	pointer to FSEDispatchRequest	Development:SIMNET:MCC: FSE:FSE.h
errCode	int	Standard C type.

Calls	
Function	**Where Described**
GetDateTime	Standard Operating System Utility function for Macintosh.
DTGElapsed	See Section 2.22.1.15.3.
ThrowDisplace	See Section 2.17.3.1.2.
UpdateBatteryDisplay	See Section 2.17.2.1.7.
UpdateBatterySelection	See Section 2.17.5.2.2.
NewPtr	Standard Memory Manager function for Macintosh.
NewHandle	Standard Memory Manager function for Macintosh.
SetUpATPRequest	See Section 2.22.1.3.2.
ATPRequest	Standard Appletalk Manager function for Macintosh.

Called By	
Function	**Where Described**
CheckTimers	See Section 2.17.5.1.1.

Table 2.17-67: LocalUnitDispatched Information.

2.17.3.2.2 LocalUnitArrived

LocalUnitArrived is called when a fire unit, specifed by *battery* and *half*, has arrived following a displacement order entered at this console. The fire unit begins setting up and the host is notified. The function call is LocalUnitArrived(battery, half). Table 2.17-68 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
now	unsigned long	Standard C type.
ab	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
req	pointer to FSEArriveRequest	Development:SIMNET:MCC: FSE:FSE.h
errCode	int	Standard C type.

Calls	
Function	**Where Described**
GetDateTime	Standard Operating System Utility function for Macintosh.
ThrowDisplace	See Section 2.17.3.1.2.
UpdateBatteryDisplay	See Section 2.17.2.1.7.
NewPtr	Standard Memory Manager function for Macintosh.
NewHandle	Standard Memory Manager function for Macintosh.
SetUpATPRequest	See Section 2.22.1.3.2.
ATPRequest	Standard Appletalk Manager function for Macintosh.
Called By	
Function	**Where Described**
CheckTimers	See Section 2.17.5.1.1.

Table 2.17-68: LocalUnitArrived Information.

2.17.3.2.3 RemoteUnitDispatched

RemoteUnitDispatched is called when a fire unit, specified by *battery* and *half*, is leaving on displacement due to an order entered at another console. The battery is withdrawn and the screen is updated. The function call is RemoteUnitDispatched(battery, half). Table 2.17-69 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
now	unsigned long	Standard C type.
delay	unsigned long	Standard C type.
Calls		
Function	**Where Described**	
WithdrawBattery	See Section 2.17.5.2.3.	
ThrowDisplace	See Section 2.17.3.1.2.	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	
UpdateBatterySelection	See Section 2.17.5.2.2.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.17.8.3.6.	

Table 2.17-69: RemoteUnitDispatched Information.

2.17.3.2.4 RemoteUnitArrived

RemoteUnitArrived is called when a fire unit has arrived following a displacement order entered at another console. The new location and azimuth of the fire unit is noted, the fire unit begins setting up, and the screen is updated. The function call is RemoteUnitArrived(battery, half, location, azimuth). Table 2.17-70 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.
location	pointer to LongPt	Development:SIMNET:libmac:longpt.h
azimuth	short	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
now	unsigned long	Standard C type.
Calls		
Function	**Where Described**	
PointToMapCoordinates	See Section 2.22.1.26.2.	
GetDateTime	Standard Operating System Utility function for Macintosh.	
ThrowDisplace	See Section 2.17.3.1.2.	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	
Called By		
Function	**Where Described**	
ProcessRequest	See Section 2.17.8.3.6.	

Table 2.17-70: RemoteUnitArrived Information.

2.17.3.2.5 InterpolatePoints

InterpolatePoints locates a point some fraction of the way between two other points. The function call is InterpolatePoints(pt1, pt2, num, denom, ptInt). Table 2.17-71 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
pt1	pointer to LongPt	Development:SIMNET:libmac:longpt.h
pt2	pointer to LongPt	Development:SIMNET:libmac:longpt.h
ptInt	pointer to LongPt	Development:SIMNET:libmac:longpt.h
num	long	Standard C type.
denom	long	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
fract	float	Standard C type.
Called By		
Function	**Where Described**	
UpdateUnitLocation	See Section 2.17.3.2.6.	

Table 2.17-71: InterpolatePoints Information.

2.17.3.2.6 UpdateUnitLocation

UpdateUnitLocation interpolates and redraws a fire unit's location. *battery* and *half* describe the fire unit. The function call is UpdateUnitLocation(battery, half). Table 2.17-72 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
now	unsigned long	Standard C type.
pt	LongPt	Development:SIMNET:libmac:longpt.h
Calls		
Function	**Where Described**	
GetDateTime	Standard OperatingSystem Utility function for Macintosh.	
InterpolatePoints	See Section 2.17.3.2.5.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
RedrawUnitLocation	See Section 2.17.3.1.9.	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	
Called By		
Function	**Where Described**	
CheckTimers	See Section 2.17.5.1.1.	

Table 2.17-72: UpdateUnitLocation Information.

2.17.4 Dialog Lists of Scheduled Missions

2.17.4.1 firetarget.c

Development:SIMNET:MCC:FSE:firetarget.c

firetarget.c implements a dialog containing a scrolling table of indirect fire targets. Table 2.17-73 describes the variables used in firetarget.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
fireTableDialogDefn	externDialogDefn	Development:SIMNET:libmac:dialog.h
fireEntryDialogDefn	externDialogDefn	Development:SIMNET:libmac:dialog.h
fireTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
fireTableColumns	extern array of ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
fireTableDialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
currentFireTarget	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
fireTargetBuffer	FireTargetDescriptor	Development:SIMNET:MCC:FSE:FSEMac.h
fireTgtNumberBuffer	int	Standard C type.
fireTableColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
fireTableField	ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
fireTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fireTableDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h
fireEntryDeleteField	PushButtonFieldDefn	Development:SIMNET:libmac:dialog.h
fireEntryNumberField	NumberFieldDefn	Development:SIMNET:libmac:dialog.h
fireEntryLocationField	CoordinatesFieldDefn	Development:SIMNET:libmac:dialog.h
fireEntryTextField	array of TextFieldDefn	Development:SIMNET:libmac:dialog.h
fireEntryFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fireEntryDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.17-73: firetarget.c Variable Information.

2.17.4.1.1 NewFireTarget

NewFireTarget allocates a record for a new fire target assignment. The function call is NewFireTarget(). Table 2.17-74 describes the internal variable used and function called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
f	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
Return Values		
Return Value	**Type**	**Meaning**
f	FireTargetHandle	New fire target handle.
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
Called By		
Function	**Where Described**	
FireEvent	See Section 2.17.1.2.5.	
ReadFireTargets	See Section 2.17.8.1.12.	
LoadCannedFireTargets	See Section 2.17.8.2.3.	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-74: NewFireTarget Information.

2.17.4.1.2 SetUpFireTable

SetUpFireTable initializes the fire target table. The function call is SetUpFireTable(). Table 2.17-75 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
port	GrafPort	Development:THINK C:Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
OpenPort	Standard Quickdraw function for Macintosh.	
TextFont	Standard Quickdraw function for Macintosh.	
TextSize	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
CharWidth	Standard Quickdraw function for Macintosh.	
ShowDialog	See Section 2.22.1.11.1.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-75: SetUpFireTable Information.

2.17.4.1.3 LookupFireTarget

LookupFireTarget looks up a fire target by its target number, *n*. The function call is LookupFireTarget(n). Table 2.17-76 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
n	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
LookupScrollTableEntry (&fireTableField, (unsigned long) n)	FireTargetHandle	The fire target specified by *n*.
Calls		
Function	**Where Described**	
LookupScrollTableEntry	See Section 2.22.1.24.1.	
Called By		
Function	**Where Described**	
ValidTgtLocation	See Section 2.17.1.6.1.	

Table 2.17-76: LookupFireTarget Information.

2.17.4.1.4 ShowFireTable

ShowFireTable makes the fire target table visible. The function call is ShowFireTable(). Table 2.17-77 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
SelectWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-77: ShowFireTable Information.

2.17.4.1.5 FireTableSelect

FireTableSelect checks the validity of the Fire Table selection and enables the menus describing the selection. The function call is FireTableSelect(defn, row, box, event). Table 2.17-78 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
SelectScrollTableEntry	See Section 2.22.1.37.1.	
EnableMenus	See Section 2.17.8.4.4.	

Table 2.17-78: FireTableSelect Information.

2.17.4.1.6 TableDrawNumber

TableDrawNumber fills in the Number field of the specified column in the Fire Target scrolling table. The function call is TableDrawNumber(defn, col, entry, box). Table 2.17-79 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 10 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	
Called By		
Function	**Where Described**	
SchedTableDrawTarget	See Section 2.17.4.3.6.	

Table 2.17-79: TableDrawNumber Information.

2.17.4.1.7 FireTableDrawLocation

FireTableDrawLocation fills in the Location field of the specified column in the Fire Target scrolling table. The function call is FireTableDrawLocation(defn, col, entry, box). Table 2.17-80 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	register pointer to char	Standard C type.
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.17-80: FireTableDrawLocation Information.

2.17.4.1.8 FireTableDrawDescription

FireTableDrawDescription fills in the Description field of the specified column in the Fire Target scrolling table. The function call is FireTableDrawDescription(defn, col, entry, box). Table 2.17-81 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	register pointer to char	Standard C type.
Calls		
Function	**Where Described**	
TextBox	Standard TextEdit function for Macintosh.	

Table 2.17-81: FireTableDrawDescription Information.

2.17.4.1.9 FireTableDrawRemarks

FireTableDrawRemarks fills in the Remarks field of the specified column in the Fire Target scrolling table. The function call is FireTableDrawRemarks(defn, col, entry, box). Table 2.17-82 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	register pointer to char	Standard C type.
Calls		
Function	**Where Described**	
TextBox	Standard TextEdit function for Macintosh.	

Table 2.17-82: FireTableDrawRemarks Information.

2.17.4.1.10 InstallFireTarget

InstallFireTarget inserts a new fire target, *f*, in the table. The function call is InstallFireTarget(f). Table 2.17-83 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
f	FireTargetHandle	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
Called By		
Function	**Where Described**	
ReadFireTargets	See Section 2.17.8.1.12.	
LoadCannedFireTargets	See Section 2.17.8.2.3.	

Table 2.17-83: InstallFireTarget Information.

2.17.4.1.11 ShowFireTarget

ShowFireTarget displays a dialog which allows editing of a fire target, *f*. The function call is ShowFireTarget (f, zoom). Table 2.17-84 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
f	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
zoom	int	Standard C type.
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ZoomScrollTableEntry	See Section 2.22.1.36.1.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
FireEvent	See Section 2.17.1.2.5.	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-84: ShowFireTarget Information.

2.17.4.1.12 FireEntryEvent

FireEntryEvent handles events in the Fire Target Entry dialog. The function call is FireEntryEvent(dialog, itemNo). Table 2.17-85 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomScrollTableEntry	See Section 2.22.1.36.1.	
UnhookFireTarget	See Section 2.17.4.1.15.	
RemoveScrollTableEntry	See Section 2.22.1.28.1	
ThrowDialog	See Section 2.22.1.11.3.	

Table 2.17-85: FireEntryEvent Information.

2.17.4.1.13 ValidNewFireTarget

ValidNewFireTarget checks the validity of the new entry in the Fire Target table. The function call is ValidNewFireTarget(dialog, fp, str). Table 2.17-86 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac: dialog.h
fp	pointer to TypeInFieldDefn	Development:SIMNET:libmac: dialog.h
str	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
e	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
b	array of 80 char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Not a valid new fire target.
1	int	Valid new fire target.
Calls		
Function	**Where Described**	
LookupScrollTableEntry	See Section 2.22.1.24.1.	
ShowCaution	See Section 2.22.1.4.1.	

Table 2.17-86: ValidNewFireTarget Information.

2.17.4.1.14 UndoFireTarget

UndoFireTarget undoes changes in the fire target entry. The function call is UndoFireTarget(). Table 2.17-87 describes the function called using this function.

Calls	
Function	**Where Described**
UpdateDialog	See Section 2.22.1.11.2.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-87: UndoFireTarget Information.

2.17.4.1.15 UnhookFireTarget

UnhookFireTarget removes all references to a fire target, *f*, which is about to be deleted. The function call is UnhookFireTarget(defn, f). Table 2.17-88 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
f	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
s	register SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
i	register short	Standard C type.
Calls		
Function	**Where Described**	
RedrawMissionIcon	See Section 2.17.2.2.5.	
Called By		
Function	**Where Described**	
FireEntryEvent	See Section 2.17.4.1.12.	
WipeFireTable	See Section 2.17.8.1.13.	

Table 2.17-88: UnhookFireTarget Information.

2.17.4.2 fpftarget.c

Development:SIMNET:MCC:FSE:fpftarget.c

fpftarget.c implements a dialog containing a scrolling table of final protective fire targets. Table 2.17-89 describes the variables used in fpftarget.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
fpfTableDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
fpfEntryDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
fpfTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
fpfTableColumns	extern array of ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
fpfTableDialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
currentFPFTarget	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
fpfTargetBuffer	FPFTargetDescriptor	Development:SIMNET:MCC:FSE:FSEMac.h
fpfTgtNumberBuffer	int	Standard C type.
fpfTableColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
fpfTableField	ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
fpfTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fpfTableDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h
fpfEntryDeleteField	PushButtonFieldDefn	Development:SIMNET:libmac:dialog.h
fpfEntryNumberField	NumberFieldDefn	Development:SIMNET:libmac:dialog.h
fpfEntryLocationFields	array of CoordinatesFieldDefn	Development:SIMNET:libmac:dialog.h
fpfEntryDescriptionField	TextFieldDefn	Development:SIMNET:libmac:dialog.h
fpfEntryFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
fpfEntryDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.17-89: fpftarget.c Variable Information.

2.17.4.2.1 NewFPFTarget

NewFPFTarget allocates a record for a new FPF target assignment. The function call is NewFPFTarget(). Table 2.17-90 describes the internal variable used and function called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
f	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
Return Values		
Return Value	**Type**	**Meaning**
f	FPFTargetHandle	New FPF target.

Calls	
Function	**Where Described**
NewHandle	Standard Memory Manager function for Macintosh.
Called By	
Function	**Where Described**
ReadFPFTargets	See Section 2.17.8.1.15.
LoadCannedFPFTargets	See Section 2.17.8.2.4.
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-90: NewFPFTarget Information.

2.17.4.2.2 SetUpFPFTable

SetUpFPFTable initializes an FPF target table. The function call is SetUpFPFTable(). Table 2.17-91 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
port	GrafPort	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
OpenPort	Standard Quickdraw function for Macintosh.	
TextFont	Standard Quickdraw function for Macintosh.	
TextSize	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
CharWidth	Standard Quickdraw function for Macintosh.	
ShowDialog	See Section 2.22.1.11.1.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-91: SetUpFPFTable Information.

2.17.4.2.3 ShowFPFTable

ShowFPFTable makes an FPF target table visible. The function call is ShowFPFTable(). Table 2.17-92 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
SelectWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-92: ShowFPFTable Information.

2.17.4.2.4 FPFTableSelect

FPFTableSelect checks for the validity of the selection and enables menus pertaining to the selection. The function call is FPFTableSelect(defn, row, box, event). Table 2.17-93 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
SelectScrollTableEntry	See Section 2.22.1.37.1.	
EnableMenus	See Section 2.17.8.4.4.	

Table 2.17-93: FPFTableSelect Information.

2.17.4.2.5 FPFTableDrawNumber

FPFTableDrawNumber fills in the Number field of the specified column in the FPF Table. The function call is FPFTableDrawNumber(defn, col, entry, box). Table 2.17-94 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	array of 255 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.17-94: FPFTableDrawNumber Information.

2.17.4.2.6 FPFTableDrawLeft

FPFTableDrawLeft fills in the Left field of the specified column in the FPF Table. The function call is FPFTableDrawLeft(defn, col, entry, box). Table 2.17-95 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	register pointer to char	Standard C type.
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.17-95: FPFTableDrawLeft Information.

2.17.4.2.7 FPFTableDrawRight

FPFTableDrawRight fills in the Right field of the specified column in the FPF Table. The function call is FPFTableDrawRight(defn, col, entry, box). Table 2.17-96 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	register pointer to char	Standard C type.
Calls		
Function	**Where Described**	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.17-96: FPFTableDrawRight Information.

2.17.4.2.8 FPFTableDrawDescription

FPFTableDrawDescription fills in the Description field of the specified column in the FPF Table. The function call is FPFTableDrawDescription(defn, col, entry, box). Table 2.17-97 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
str	register pointer to char	Standard C type.
Calls		
Function	**Where Described**	
TextBox	Standard TextEdit function for Macintosh.	

Table 2.17-97: FPFTableDrawDescription Information.

2.17.4.2.9 InstallFPFTarget

InstallFPFTarget inserts a new FPF target, *f*, in the table. The function call is InstallFPFTarget(f). Table 2.17-98 describes the parameter used and function called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
f	FPFTargetHandle	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
Called By		
Function	**Where Described**	
ReadFPFTargets	See Section 2.17.8.1.15.	
LoadCannedFPFTargets	See Section 2.17.8.2.4.	

Table 2.17-98: InstallFPFTarget Information.

2.17.4.2.10 ShowFPFTarget

ShowFPFTarget displays a dialog which allows the editing of an FPF target, *f*. The function call is ShowFPFTarget(f, zoom). Table 2.17-99 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
f	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
zoom	int	Standard C type.
Calls		
Function	**Where Described**	
ShowDialog	See Section 2.22.1.11.1.	
ZoomScrollTableEntry	See Section 2.22.1.36.1.	
ShowWindow	Standard Window Manager function for Macintosh.	
Called By		
Function	**Where Described**	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-99: ShowFPFTarget Information.

2.17.4.2.11 FPFEntryEvent

FPFEntryEvent handles events in the FPF dialog. The function call is FPFEntryEvent(dialog, itemNo). Table 2.17-100 describes the parameters used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomScrollTableEntry	See Section 2.22.1.37.1.	
RemoveScrollTableEntry	See Section 2.22.1.28.1.	
ThrowDialog	See Section 2.22.1.11.3.	

Table 2.17-100: FPFEntryEvent Information.

2.17.4.2.12 ValidNewFPFTarget

ValidNewFPFTarget checks the validity of the new FPF Target entry. The function call is ValidNewFPFTarget(dialog, fp, str). Table 2.17-101 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:scroll.h
fp	pointer to TypeInFieldDefn	Development:SIMNET:libmac:scroll.h
str	pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
e	register ScrollTableEntryHandle	Development:SIMNET:libmac:scroll.h
b	array of 80 char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	New FPF target not valid.
1	int	New FPF target valid.
Calls		
Function	**Where Described**	
LookupScrollTableEntry	See Section 2.22.1.24.1.	
ShowCaution	See Section 2.22.1.4.1.	

Table 2.17-101: ValidNewFPFTarget Information.

2.17.4.2.13 UndoFPFTarget

UndoFPFTarget undoes any changes in the FPF target entry. The function call is UndoFPFTarget(). Table 2.17-102 describes the function called using this function.

Calls	
Function	**Where Described**
UpdateDialog	See Section 2.22.1.11.2.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-102: UndoFPFTarget Information.

2.17.4.3 schedule.c

Development:SIMNET:MCC:FSE:schedule.c

schedule.c implements a dialog containing a scrolling table of prescheduled fire missions and controls the initiation of prescheduled missions. Table 2.17-103 describes the variables used in schedule.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
schedTableDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
schedEntryDialogDefn	extern DialogDefn	Development:SIMNET:libmac:dialog.h
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
schedTableColumns	extern array of ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
schedTableDialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
currentSchedMission	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
schedMissionBuffer	SchedMissionDescriptor	Development:SIMNET:MCC:FSE:FSEMac.h
schedTargetBuffer	TargetLocationBuffer	Development:SIMNET:MCC:FSE:FSEMac.h
schedTableColumns	array of ScrollTableColumnDefn	Development:SIMNET:libmac:scroll.h
schedTableField	ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
schedTableFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
schedTableDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h
schedDeleteField	PushButtonFieldDefn	Development:SIMNET:libmac:dialog.h
schedTimeField	DTGFieldDefn	Development:SIMNET:libmac:dialog.h
schedTargetField	TextFieldDefn	Development:SIMNET:libmac:dialog.h
schedAmmoFields	array of RBFieldDefn	Development:SIMNET:libmac:dialog.h
schedBatteryFields	array of CBFieldDefn	Development:SIMNET:libmac:dialog.h
schedRoundsField	NumberFieldDefn	Development:SIMNET:libmac:dialog.h
schedEntryFieldList	pointer to array of FieldDefn	Development:SIMNET:libmac:dialog.h
schedEntryDialogDefn	DialogDefn	Development:SIMNET:libmac:dialog.h

Table 2.17-103: schedule.c Variable Information.

2.17.4.3.1 NewSchedMission

NewSchedMission allocates a record for a new scheduled fire mission. The function call is NewSchedMission(). Table 2.17-104 describes the internal variables used and function called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
s	register SchedMission Handle	Development:SIMNET:MCC: FSE:FSEMac.h
i	short	Standard C type.
j	short	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
s	SchedMissionHandle	New scheduled fire mission.
Calls		
Function	**Where Described**	
NewHandle	Standard Memory Manager function for Macintosh.	
Called By		
Function	**Where Described**	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-104: NewSchedMission Information.

2.17.4.3.2 SetUpSchedTable

SetUpSchedTable initializes a fire mission schedule. The function call is SetUpSchedTable(). Table 2.17-105 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
port	GrafPort	Development:THINK C: Mac #includes:Quickdraw.h
Calls		
Function	**Where Described**	
OpenPort	Standard Quickdraw function for Macintosh.	
TextFont	Standard Quickdraw function for Macintosh.	
TextSize	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
ShowDialog	See Section 2.22.1.11.1.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-105: SetUpSchedTable Information.

2.17.4.3.3 ShowSchedTable

ShowSchedTable makes a fire mission schedule visible. The function call is ShowSchedTable(). Table 2.17-106 describes the functions called using this function.

Calls	
Function	**Where Described**
ShowWindow	Standard Window Manager function for Macintosh.
SelectWindow	Standard Window Manager function for Macintosh.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-106: ShowSchedTable Information.

2.17.4.3.4 SchedTableSelect

SchedTableSelect checks for validity of theSchedule Table selection. The function call is SchedTableSelect(defn, row, box, event). Table 2.17-107 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
row	int	Standard C type.
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
event	pointer to EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
SelectScrollTableEntry	See Section 2.22.1.37.1.	
EnableMenus	See Section 2.17.8.4.4.	

Table 2.17-107: SchedTableSelect Information.

2.17.4.3.5 SchedTableDrawTime

SchedTableDrawTime fills in the Time field of the specified column in the Schedule Table. The function call is SchedTableDrawTime (defn, col, entry, box). Table 2.17-108 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
s	register SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h
str	array of 20 char	Standard C type.
Calls		
Function	**Where Described**	
DTGToString	See Section 2.22.1.15.1.	
DrawText	Standard Quickdraw function for Macintosh.	

Table 2.17-108: SchedTableDrawTime Information.

2.17.4.3.6 SchedTableDrawTarget

SchedTableDrawTarget fills in the Target field of the specified column in the Schedule Table. The function call is SchedTableDrawTarget(defn, col, entry, box). Table 2.17-109 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
s	register SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h

Calls	
Function	**Where Described**
TableDrawNumber	See Section 2.17.4.1.6.
DrawText	Standard Quickdraw function for Macintosh.

Table 2.17-109: SchedTableDrawTarget Information.

2.17.4.3.7 SchedTableDrawDescription

SchedTableDrawDescription fills in the Description field of the specified column in the Schedule Table. The function call is SchedTableDrawDescription (defn, col, entry, box). Table 2.17-110 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
s	register SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h
str	register pointer to char	Standard C type.
Calls		
Function	**Where Described**	
TextBox	Standard TextEdit function for Macintosh.	

Table 2.17-110: SchedTableDrawDescription Information.

2.17.4.3.8 SchedTableDrawUnits

SchedTableDrawUnits fills in the Units field of the specified column in the Schedule Table. The function call is SchedTableDrawUnits(defn, col, entry, box). Table 2.17-111 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	register ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
s	register SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h
i	register short	Standard C type.
batteryTypeName	pointer to array of char	Standard C type.
batteryType	array of numberBatteryTypes char	Standard C type.
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.17-111: SchedTableDrawUnits Information.

2.17.4.3.9 SchedTableDrawRounds

SchedTableDrawRounds fills in the Rounds field of the specified column in the Schedule Table. The function call is SchedTableDrawRounds(defn, col, entry, box). Table 2.17-112 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
s	register SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h
str	array of 10 char	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
Move	Standard Quickdraw function for Macintosh.	
StringWidth	Standard Quickdraw function for Macintosh.	
DrawString	Standard Quickdraw function for Macintosh.	

Table 2.17-112: SchedTableDrawRounds Information.

2.17.4.3.10 SchedTableDrawStatus

SchedTableDrawStatus fills in the Status field of the specified column in the Schedule Table. The function call is SchedTableDrawStatus(defn, col, entry, box). Table 2.17-113 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
col	pointer to ScrollTableColumnDefn	Development:SIMNET:libmac: scroll.h
entry	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h
box	pointer to Rect	Development:THINK C: Mac #includes:MacTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
s	register SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h
statusString	pointer to array of char	Standard C type.
Calls		
Function	**Where Described**	
DrawString	Standard Quickdraw function for Macintosh.	
TextFace	Standard Quickdraw function for Macintosh.	

Table 2.17-113: SchedTableDrawStatus Information.

2.17.4.3.11 ShowSchedMission

ShowSchedMission displays a dialog which allows editing of a scheduled mission, *s*. The function call is ShowSchedMission(s, zoom). Table 2.17-114 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
s	SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h
zoom	int	Standard C type.
Calls		
Function	**Where Described**	
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.	
ShowDialog	See Section 2.22.1.11.1.	
ZoomScrollTableEntry	See Section 2.22.1.36.1.	
ShowWindow	Standard Window Manager function for Macintosh.	
EstablishBatterySelection	See Section 2.17.5.2.1.	
Called By		
Function	**Where Described**	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-114: ShowSchedMission Information.

2.17.4.3.12 SchedEntryEvent

SchedEntryEvent handles events in the Schedule Entry dialog. The function call is SchedEntryEvent(dialog, itemNo). Table 2.17-115 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
itemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	short	Standard C type.
j	short	Standard C type.
batteriesChosen	short	Standard C type.
Calls		
Function	**Where Described**	
CheckMandatoryFields	See Section 2.22.1.13.1.	
ShowCaution	See Section 2.22.1.4.1.	
InstallScrollTableEntry	See Section 2.22.1.21.1.	
HideWindow	Standard Window Manager function for Macintosh.	
ZoomScrollTableEntry	See Section 2.22.1.36.1.	
RemoveScrollTableEntry	See Section 2.22.1.28.1.	
ThrowDialog	See Section 2.22.1.11.2.	

Table 2.17-115: SchedEntryEvent Information.

2.17.4.3.13 ValidTime

ValidTime checks to see if a valid time has been entered in the Schedule Table. The function call is ValidTime(dialog, fp, str). Table 2.17-116 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
fp	pointer to DTGFieldDefn	Development:SIMNET:libmac:dialog.h
str	register pointer to char	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
now	unsigned long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	No valid time entered.
1	int	Valid time entered.

Calls	
Function	**Where Described**
GetDateTime	Standard Operating System Utility function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.

Table 2.17-116: ValidTime Information.

2.17.4.3.14 UndoSchedMission

UndoSchedMission undoes changes in a scheduled fire mission entry. The function call is UndoSchedMission(). Table 2.17-117 describes the functions called using this function.

Calls	
Function	**Where Described**
NumToString	Standard Binary to Decimal Conversion Package function for Macintosh.
UpdateDialog	See Section 2.22.1.11.2.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-117: UndoSchedMission Information.

2.17.4.3.15 StartSchedMission

StartSchedMission is called when a scheduled fire mission, *s*, is due to begin. The routine checks that the mission is not being edited and that all the requested guns are available. An unused mission descriptor is found and filled in. The routine checks that the guns have ammo and can reach the target before they are assigned to the mission. The guns are fired, and the mission schedule is updated to show that the mission is in progress. The function call is StartSchedMission(s). Table 2.17-118 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
s	register SchedMissionHandle	Development:SIMNET:MCC: FSE:FSEMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
i	short	Standard C type.
j	short	Standard C type.
str	array of 100 char	Standard C type.
reason	pointer to char	Standard C type.

Calls	
Function	**Where Described**
ShowCaution	See Section 2.22.1.4.1.
SetSchedMission	See Section 2.17.4.3.16.
BatterySelectionViable	See Section 2.17.5.1.9.
EndMission	See Section 2.17.8.5.2.
ApplyToBatteries	See Section 2.17.5.1.5.
UpdateBatterySelection	See Section 2.17.5.2.2.
FireVolley	See Section 2.17.5.1.2.
UpdateMissionDisplay	See Section 2.17.8.5.5.
Called By	
Function	**Where Described**
CheckTimers	See Section 2.17.5.1.1.

Table 2.17-118: StartSchedMission Information.

2.17.4.3.16 SetSchedMission

SetSchedMission sets the status, *status*, of a scheduled mission, *s*. The function call is SetSchedMission(s, status). Table 2.17-119 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
s	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
status	int	Standard C type.
Calls		
Function	**Where Described**	
UpdateScrollTableEntry	See Section 2.22.1.31.1.	
EnableMenus	See Section 2.17.8.4.4.	
Called By		
Function	**Where Described**	
StartSchedMission	See Section 2.17.4.3.15.	
CheckFiring	See Section 2.17.1.3.6.	
FireVolley	See Section 2.17.5.1.2.	
ReadSchedMissions	See Section 2.17.8.1.18.	

Table 2.17-119: SetSchedMission Information.

2.17.5 Model Operation of Artillery Batteries

2.17.5.1 model.c
Development:SIMNET:MCC:FSE:model.c

model.c contains routines that model the firing of shells by artillery batteries. Table 2.17-120 describes the variables used in model.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h

Table 2.17-120: model.c Variable Information.

2.17.5.1.1 CheckTimers

CheckTimers looks for any time-driven changes in status. The routine checks for any missions which are ready to fire another round or report "splash". Scheduled fire missions which are due to begin are checked. Positions of displacing fire units are updated. The function call is CheckTimers(). Table 2.17-121 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
s	register SchedMisssionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
t	register unsigned long	Standard C type.
now	unsigned long	Standard C type.
mission	short	Standard C type.
battery	short	Standard C type.
half	short	Standard C type.
Calls		
Function	**Where Described**	
GetDateTime	Standard Operating System Utility function for Macintosh.	
FireVolley	See Section 2.17.5.1.2.	
UpdateMissionState	See Section 2.17.8.5.6.	
StartSchedMission	See Section 2.17.4.3.15.	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	
UpdateBatterySelection	See Section 2.17.5.2.2.	
LocalUnitDispatched	See Section 2.17.3.2.1.	
LocalUnitArrived	See Section 2.17.3.2.2.	
UpdateUnitLocation	See Section 2.17.3.2.6.	

Called By	
Function	**Where Described**
MainEventLoop	See Section 2.17.8.3.3.

Table 2.17-121: CheckTimers Information.

2.17.5.1.2 FireVolley

FireVolley fires a volley for a mission, *m*. Volleys are fired on a per-battery basis. If the batter is not participating, it is ignored. If firing FPF, the fuze that the battery has most of is chosen. Each battery half assigned to the mission is intialized on the first volley, then checked on subsequent volleys. The function call is FireVolley(m, first). Table 2.17-122 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
first	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
b	register pointer to Battery	Development:SIMNET:MCC:FSE:FSEMac.h
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
now	unsigned long	Standard C type.
newReadyTime	unsigned long	Standard C type.
newSplashTime	unsigned long	Standard C type.
t	unsigned long	Standard C type.
i	short	Standard C type.
j	short	Standard C type.
ammo	short	Standard C type.
halfFired	short	Standard C type.
target	pointer to LongPt	Development:SIMNET:libmac:longpt.h
perf	pointer to Performance	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
GetDateTime	Standard Operating System Utility function for Macintosh.	
TimeOfFlight	See Section 2.17.5.1.11.	
SendFireRequest	See Section 2.17.5.1.3.	
SendFPFRequest	See Section 2.17.5.1.4.	
UpdateAmmoDisplay	See Section 2.17.2.1.8.	
SetSchedMission	See Section 2.17.4.3.16.	
EndFireMission	See Section 2.17.1.2.10.	
UpdateBatterySelection	See Section 2.17.5.2.2.	
UpdateMissionDisplay	See Section 2.17.8.5.5.	

Called By	
Function	**Where Described**
CheckTimers	See Section 2.17.5.1.1.
AdjustEvent	See Section 2.17.1.1.2.
FFEEvent	See Section 2.17.1.3.3.
CommenceFFE	See Section 2.17.1.3.4.
FPFEvent	See Section 2.17.1.4.6.
StartSchedMission	See Section 2.17.4.3.15.

Table 2.17-122: FireVolley Information.

2.17.5.1.3 SendFireRequest

SendFireRequest tells the host of a fire for effect or an adjust fire. This function is only compiled if VERSION is APPLETALK. The function call is SendFireRequest(m, b, half, ammo). Table 2.17-123 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
b	pointer to Battery	Development:SIMNET:MCC: FSE:FSEMac.h
half	int	Standard C type.
first	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ab	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
req	pointer to FSEFireRequest	Development:SIMNET:MCC: FSE:FSEMac.h
errCode	int	Standard C type.
Calls		
Function	**Where Described**	
NewPtr	Standard Memory Manager function for Macintosh.	
NewHandle	Standard Memory Manager function for Macintosh.	
SetUpATPRequest	See Section 2.17.1.3.2.	
ATPRequest	Standard Appletalk Manager function for Macintosh.	
Called By		
Function	**Where Described**	
FireVolley	See Section 2.17.5.1.2.	

Table 2.17-123: SendFireRequest Information.

2.17.5.1.4 SendFPFRequest

SendFPFRequest tells the host of a final protective fire volley. This function is only compiled if VERSION is APPLETALK. The function call is SendFPFRequest(m, b, half, ammo). Table 2.17-124 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
b	pointer to Battery	Development:SIMNET:MCC: FSE:FSEMac.h
half	int	Standard C type.
ammo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ab	ABRecHandle	Development:THINK C: Mac #includes:Appletalk.h
req	pointer to FSEFPFRequest	Development:SIMNET:MCC: FSE:FSEMac.h
errCode	int	Standard C type.
Calls		
Function	**Where Described**	
NewPtr	Standard Memory Manager function for Macintosh.	
NewHandle	Standard Memory Manager function for Macintosh.	
SetUpATPRequest	See Section 2.22.1.3.2.	
ATPRequest	Standard Appletalk Manager function for Macintosh.	
Called By		
Function	**Where Described**	
FireVolley	See Section 2.17.5.1.2.	

Table 2.17-124: SendFPFRequest Information.

2.17.5.1.5 ApplyToBatteries

ApplyToBatteries calls a function, *f*, for each battery half being used by a mission, *m*. The function call is ApplyToBatteries(f, m). Table 2.17-125 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
(f)()	pointer to function returning int	Standard C type.
m	pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	short	Standard C type.
j	short	Standard C type.
Called By		
Function	**Where Described**	
BatterySelectionViable	See Section 2.17.5.1.9.	
EndFireMission	See Section 2.17.1.2.10.	
CheckFiring	See Section 2.17.1.3.6.	
FPFEvent	See Section 2.17.1.4.6.	
UpdateObserver	See Section 2.17.8.5.7.	
StartSchedMission	See Section 2.17.4.3.15.	

Table 2.17-125: ApplyToBatteries Information.

2.17.5.1.6 StartMissionBattery

StartMissionBattery begins a battery's involvement in a mission, *m*. *b* and *half* describe the battery. The function call is StartMissionBattery(b, half, m). Table 2.17-126 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.
half	int	Standard C type.
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	
Called By		
Function	**Where Described**	
ToggleBattery	See Section 2.17.8.5.4.	

Table 2.17-126: StartMissionBattery Information.

2.17.5.1.7 EndMissionBattery

EndMissionBattery terminates a battery's involvement in a mission, *m*. *b* and *half* describe the battery. The function call is EndMissionBattery(b, half, m). Table 2.17-127 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.
half	int	Standard C type.
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC: FSE:FSEMac.h
Calls		
Function	**Where Described**	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	
Called By		
Function	**Where Described**	
ToggleBattery	See Section 2.17.8.5.4.	

Table 2.17-127: EndMissionBattery Information.

2.17.5.1.8 CheckBattery

CheckBattery stops a battery from firing. The function call is CheckBattery(b, half, m). Table 2.17-128 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.
half	int	Standard C type.
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h

Table 2.17-128: CheckBattery Information.

2.17.5.1.9 BatterySelectionViable

BatterySelectionViable ensures that some battery has been chosen for a given mission, *m*, that the chosen batteries have some quantity of the requested ammunition, that the chosen batteries are within range of the target and that only the mortars are selected to fire white phosphorus rounds. ApplyToBatteries uses the function BatteryViable, described below, to determine if a given battery is viable. The function call is BatterySelectionViable(m). Table 2.17-129 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC: FSE:FSEMac.h

Return Values		
Return Value	**Type**	**Meaning**
batteryViable	int	0 if battery not viable, 1 if it is.
Calls		
Function	**Where Described**	
ApplyToBatteries	See Section 2.17.5.1.5.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
ShowAdjust	See Section 2.17.1.1.1.	
ShowFFE	See Section 2.17.1.3.1.	
FPFEvent	See Section 2.17.1.4.6.	
StartSchedMission	See Section 2.17.4.3.15.	

Table 2.17-129: BatterySelectionViable Information.

2.17.5.1.10 BatteryViable

BatteryViable determines if a given battery, specified by *b* and *half*, is viable for a given mission, *m*. The function call is BatteryViable(b, half, m). Table 2.17-130 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.
half	int	Standard C type.
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
distanceCheat	extern short	Standard C type.
bty	register pointer to Battery	Development:SIMNET:MCC:FSE:FSEMac.h
hp	pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
pt	pointer to LongPt	Development:SIMNET:libmac:longpt.h
range	long	Standard C type.
str	array of 256 char	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
0	int	Battery not viable.
1	int	Battery viable.

Calls	
Function	**Where Described**
ShowCaution	See Section 2.22.1.4.1.
DistBetween2Pts	See Section2.22.1.23.2.

Table 2.17-130: BatteryViable Information.

2.17.5.1.11 TimeOfFlight

TimeOfFlight returns the time of flight, in seconds, for a round, described by *b*, *half* and *ammo*, to reach a target location, *location*, from a particular battery. The function call is TimeOfFlight(b, half, location, ammo). Table 2.17-131 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	pointer to Battery	Development:SIMNET:MCC:FSE:FSEMac.h
half	int	Standard C type.
location	pointer to LongPt	Development:SIMNET:libmac:longpt.h
ammo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
pt	pointer to LongPt	Development:SIMNET:libmac:longpt.h
d	long	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
b->perf->timeOfFlight[d]	int	Time of flight in seconds.
Calls		
Function	**Where Described**	
DistBetween2Pts	See Section 2.22.1.23.2.	
Called By		
Function	**Where Described**	
FireVolley	See Section 2.17.5.1.2.	

Table 2.17-131: TimeOfFlight Information.

2.17.5.2 units.c

units.c contains routines that control the assignment of artillery batteries to indirect fire missions.

2.17.5.2.1 EstablishBatterySelection

EstablishBatterySelection adjusts a matrix of fire unit selection checkboxes to represent only the batteries in existence. The function call is EstablishBatterySelection(dialog, firstItemNo). Table 2.17-132 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	DialogPtr	Development:THINK C: Mac #includes:DialogMgr.h
firstItemNo	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	int	Standard C type.
j	int	Standard C type.
itemNo	int	Standard C type.
itemType	int	Standard C type.
battery	register pointer to Battery	Development:SIMNET:MCC: FSE:FSEMac.h
item	Handle	Development:THINK C: Mac #includes:MacTypes.h
box	Rect	Development:THINK C: Mac #includes:MacTypes.h
pStr	array of 256 char	Standard C type.
Calls		
Function	**Where Described**	
GetDItem	Standard Dialog Manager function for Macintosh.	
SetIText	Standard Dialog Manager function for Macintosh.	
SetCTitle	Standard Control Manager function for Macintosh.	
OffsetRect	Standard Quickdraw function for Macintosh.	
SetDItem	Standard Dialog Manager function for Macintosh.	
HideControl	Standard Control Manager function for Macintosh.	
Called By		
Function	**Where Described**	
SetUpFireMission	See Section 2.17.1.2.1.	
SetUpFPFMission	See Section 2.17.1.4.1.	
ShowSchedMission	See Section 2.17.4.3.11.	

Table 2.17-132: EstablishBatterySelection Information.

2.17.5.2.2 UpdateBatterySelection

UpdateBatterySelection updates the display of the selection of batteries for performing a fire mission. Note that this routine embeds assumptions about the order of battery selection items within fire dialogs. The function call is UpdateBatterySelection(). Table 2.17-133 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
j	register short	Standard C type.
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h
dialog	DialogPtr	Development:THINK C:Mac #includes:DialogMgr.h
firstItemNo	int	Standard C type.
itemNo	int	Standard C type.

Calls	
Function	**Where Described**
GetFireDialogInfo	See Section 2.17.1.2.2.
GetFPFDialogInfo	See Section 2.17.1.4.2.
DisableControl	See Section 2.22.1.7.1.
EnableControl	See Section 2.22.1.7.2.

Called By	
Function	**Where Described**
ShowFireMission	See Section 2.17.1.2.4.
ShowFPFMission	See Section 2.17.1.4.5.
LocalUnitDispatched	See Section 2.17.3.2.1.
RemoteUnitDispatched	See Section 2.17.3.2.3.
StartSchedMission	See Section 2.17.4.3.15.
CheckTimers	See Section 2.17.5.1.1.
FireVolley	See Section 2.17.5.1.2.
UpdateMissionDisplay	See Section 2.17.8.5.5.

Table 2.17-133: UpdateBatterySelection Information.

2.17.5.2.3 WithdrawBattery

WithdrawBattery withdraws a fire unit from any mission in which it is participating. The function call is WithdrawBattery(battery, half, newState). Table 2.17-134 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
battery	int	Standard C type.
half	int	Standard C type.
newState	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
hp	register pointer to struct Half	Development:SIMNET:MCC:FSE:FSEMac.h

Calls	
Function	**Where Described**
UpdateMissionDisplay	See Section 2.17.8.5.5.

Called By	
Function	**Where Described**
RemoteUnitDispatched	See Section 2.17.3.2.3.
ProcessRequest	See Section 2.17.8.3.6.

Table 2.17-134: WithdrawBattery Information.

2.17.6 Appearance of FSE User Interface

2.17.6.1 FSE Pictures
Development:SIMNET:MCC:FSE:FSE Pictures

FSE Pictures contains resources that determine the appearance of the FSE application's user interface.

2.17.6.2 resource.h
Development:SIMNET:MCC:FSE:resource.h

resource.h defines the numbers of the resources present in the FSE Pictures resource file.

2.17.7 FSE Console Definitions

2.17.7.1 FSE.h
Development:SIMNET:MCC:FSE:FSE.h

FSE.h defines the representation of information communicated between the FSE Macintosh application and the MCC host.

2.17.7.2 FSEMac.h
Development:SIMNET:MCC:FSE:FSEMac.h

FSEMac.h defines types, constants and routines used within the FSE Macintosh application. Table 2.17-135 describes the variables used in FSEMac.h.

Variables		
Variable	**Type**	**Where Typedef Declared**
missions	extern array of Mission	Development:SIMNET:MCC:FSE:FSEMac.h
performance	extern array of Performance	Development:SIMNET:MCC:FSE:FSEMac.h
halfName[][2]	extern pointer to matrix of char	Standard C type.
batteries	extern array of Battery	Development:SIMNET:MCC:FSE:FSEMac.h
screenState	extern short	Standard C type.
popUpDialog	extern pointer to DialogState	Development:SIMNET:libmac:dialog.h
entryDialog	extern pointer to DialogState	Development:SIMNET:libmac:dialog.h
currentMission	extern pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h

Table 2.17-135: FSEMac.h Variable Information.

2.17.7.3 version.h

Development:SIMNET:MCC:FSE:version.h

version.h defines constants that determine which version of the FSE application is compiled.

2.17.7.4 data.c
Development:SIMNET:MCC:FSE:data.c

data.c defines various data structures used by the FSE application, such as tables of information about the performance of artillery batteries. Table 2.17-136 describes the variables used in data.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
application	pointer to char	Standard C type.
authors	pointer to char	Standard C type.
copyright	pointer to char	Standard C type.
mortarPerformance	Performance	Development:SIMNET:MCC:FSE:FSEMac.h
howitzerPerformance	Performance	Development:SIMNET:MCC:FSE:FSEMac.h
halfName[][2]	pointer to matrix char	Standard C type.
batteries	array of maxNumberBatteries Battery	Development:SIMNET:MCC:FSE:FSEMac.h
missions	array of maxNumberMissions Mission	Development:SIMNET:MCC:FSE:FSEMac.h
screenState	short	Standard C type.
popUpDialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
entryDialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
currentMission	pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
distanceCheat	short	Standard C type.

Table 2.17-136: data.c Variable Information.

2.17.8 FSE Console Software

2.17.8.1 file.c
Development:SIMNET:MCC:FSE:file.c

file.c contains routines for creating and reading Macintosh files holding lists of indirect fire targets and scheduled missions. Table 2.17-137 describes the variables used in file.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
badMissions	pointer to SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
errStr	array of 128 char	Standard C type.
bytesOut	int	Standard C type.

Table 2.17-137: file.c Variable Information.

2.17.8.1.1 NewPresets

NewPresets allows the user to set up new preset data in the files holding the indirect fire target lists and scheduled mission lists. The function call is NewPresets(). Table 2.17-138 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
Calls		
Function	**Where Described**	
GetConfig	See Section 2.17.8.1.4.	
WipeFireTable	See Section 2.17.8.1.13.	
WipeFPFTable	See Section 2.17.8.1.16.	
WipeSchedTable	See Section 2.17.8.1.20.	
ReadTerrainMap	See Section 2.17.8.1.8.	
Called By		
Function	**Where Described**	
MenuCommand	See Section 2.17.8.4.5.	

Table 2.17-138: NewPresets Information.

2.17.8.1.2 LoadPresets

LoadPresets reads in the preset target files which contain canned target and scheduled mission data. The function call is LoadPresets(). Table 2.17-139 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
where	Point	Development:THINK C: Mac #includes:MacTypes.h
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	FInfo	Development:THINK C: Mac #includes:FileMgr.h
theFile	*int*	Standard C type.
theTypeList	SFTypeList	Development:THINK C: Mac #includes:StdFilePkg.h

Calls	
Function	**Where Described**
SFGetFile	Standard Standard File Package function for Macintosh.
FSOpen	Standard File Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
GetConfig	See Section 2.17.8.1.4.
CheckConfig	See Section 2.17.8.1.6.
WipeFireTable	See Section 2.17.8.1.13.
WipeFPFTable	See Section 2.17.8.1.16.
WipeSchedTable	See Section 2.17.8.1.20.
ReadTerrainMap	See Section 2.17.8.1.8.
ReadFireTargets	See Section 2.17.8.1.12.
ReadFPFTargets	See Section 2.17.8.1.15.
ReadSchedMissions	See Section 2.17.8.1.18.
FSClose	Standard File Manager function for Macintosh.
FlushVol	Standard File Manager function for Macintosh.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-139: LoadPresets Information.

2.17.8.1.3 SavePresets

SavePresets saves a preset file containing Fire Target, FPFTarget, and Scheduled Mission data. The function call is SavePresets(). Table 2.17-140 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
where	Point	Development:THINK C: Mac #includes:MacTypes.h
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	FInfo	Development:THINK C: Mac #includes:FileMgr.h
theFile	int	Standard C type.

Calls	
Function	**Where Described**
SFPutFile	Standard Standard File Package function for Macintosh.
GetFInfo	Standard File Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
Create	Standard File Manager function for Macintosh.
FSOpen	Standard File Manager function for Macintosh.
GetConfig	See Section 2.17.8.1.4.
SaveConfig	See Section 2.17.8.1.5.
SaveFireTargets	See Section 2.17.8.1.11.
SaveFPFTargets	See Section 2.17.8.1.14.
SaveSchedMissions	See Section 2.17.8.1.17.
SetEOF	Standard File Manager function for Macintosh.
FSClose	Standard File Manager function for Macintosh.
FlushVol	Standard File Manager function for Macintosh.
Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-140: SavePresets Information.

2.17.8.1.4 GetConfig

GetConfig lists the configuration options database version and name for the preset file. The function call is GetConfig(). Table 2.17-141 describes the functions calling this function.

Called By	
Function	**Where Described**
NewPresets	See Section 2.17.8.1.1.
LoadPresets	See Section 2.17.8.1.2.
SavePresets	See Section 2.17.8.1.3.

Table 2.17-141: GetConfig Information.

2.17.8.1.5 SaveConfig

SaveConfig saves the configuration of the preset file specified by the parameter *theFile*. This routine is called by SavePresets() to write out the file. The function call is SaveConfig(theFile). Table 2.17-142 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
count	long	Standard C type.
configSize	int	Standard C type.
Return Values		
Return Value	**Type**	**Meaning**
-1	int	Write error in function.
Calls		
Function	**Where Described**	
FSWrite	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
SavePresets	See Section 2.17.8.1.3.	

Table 2.17-142: SaveConfig Information.

2.17.8.1.6 CheckConfig

CheckConfig checks the validity of the preset file specified by the parameter *theFile*. This routine is called by LoadPresets() to read in the preset file. The function call is CheckConfig(theFile). Table 2.17-143 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
count	long	Standard C type.
configSize	int	Standard C type.
resCode	int	Standard C type.
configFile	struct config	Development:SIMNET:MCC: FSE:file.c
Calls		
Function	**Where Described**	
FSRead	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
LoadPresets	See Section 2.17.8.1.2.	

Table 2.17-143: CheckConfig Information.

2.17.8.1.7 SaveTerrainMap

SaveTerrainMap saves terrain map data to the file specified by *theFile*. The function call is SaveTerrainMap(). Table 2.17-144 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
mapSize	int	Standard C type.
count	long	Standard C type.
astr	array of 128 char	Standard C type.
where	Point	Development:THINK C: Mac #includes:MacTypes.h
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	FInfo	Development:THINK C: Mac #includes:FileMgr.h
theFile	int	Standard C type.
tbytesOut	int	Standard C type.
Calls		
Function	**Where Described**	
SFPutFile	Standard Standard File Package function for Macintosh.	
GetFInfo	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
Create	Standard File Manager function for Macintosh.	
FSOpen	Standard File Manager function for Macintosh.	
FSWrite	Standard File Manager function for Macintosh.	
SetEOF	Standard File Manager function for Macintosh.	
FSClose	Standard File Manager function for Macintosh.	
FlushVol	Standard File Manager function for Macintosh.	
Called By		
Function	**Where Described**	
MainEventLoop	See Section 2.17.8.3.3.	

Table 2.17-144: SaveTerrainMap Information.

2.17.8.1.8 ReadTerrainMap

ReadTerrainMap reads the data file containing the terrain map. The function call is ReadTerrainMap(query). Table 2.17-145 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
query	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
count	long	Standard C type.
mapSize	int	Standard C type.
theReply	SFReply	Development:THINK C: Mac #includes:StdFilePkg.h
resCode	OSErr	Development:THINK C: Mac #includes:MacTypes.h
theInfo	FInfo	Development:THINK C: Mac #includes:FileMgr.h
theTypeList	SFTypeList	Development:THINK C: Mac #includes:StdFilePkg.h
vRefNum	int	Standard C type.
driveNum	int	Standard C type.
theFile	int	Standard C type.
freeBytes	long	Standard C type.
volName	Str255	Development:THINK C: Mac #includes:MacTypes.h
where	Point	Development:THINK C: Mac #includes:MacTypes.h

Return Values		
Return Value	**Type**	**Meaning**
-1	int	Unsuccessful.

Calls	
Function	**Where Described**
ShowCaution	See Section 2.22.1.4.1.
SFGetFile	Standard Standard File Package function for Macintosh.
FSOpen	Standard File Manager function for Macintosh.
GetVol	Standard File Manager function for Macintosh.
FSRead	Standard File Manager function for Macintosh.
FSClose	Standard File Manager function for Macintosh.
FlushVol	Standard File Manager function for Macintosh.

Called By	
Function	**Where Described**
NewPresets	See Section 2.17.8.1.1.
LoadPresets	See Section 2.17.8.1.2.

Table 2.17-145: ReadTerrainMap Information.

2.17.8.1.9 SaveStatus

SaveStatus saves the status of the mortar batteries to the data file specified by the parameter *theFile*. The function call is SaveStatus(theFile). Table 2.17-146 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
count	long	Standard C type.
astr	array of 128 char	Standard C type.
batsize	int	Standard C type.
missSize	int	Standard C type.
Calls		
Function	**Where Described**	
FSWrite	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	

Table 2.17-146: SaveStatus Information.

2.17.8.1.10 ReadStatus

ReadStatus reads the status of the mortar batteries from the data file specified by the parameter *theFile*. The function call is ReadStatus(theFile). Table 2.17-147 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
count	long	Standard C type.
batSize	int	Standard C type.
missSize	int	Standard C type.
resCode	int	Standard C type.
Calls		
Function	**Where Described**	
FSRead	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	

Table 2.17-147: ReadStatus Information.

2.17.8.1.11 SaveFireTargets

SaveFireTargets saves Fire Target data to the data file specified by the parameter *theFile*. The function call is SaveFireTargets(theFile). Table 2.17-148 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
ft	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numTargets	int	Standard C type.
i	int	Standard C type.
fireTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
FSWrite	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
SavePresets	See Section 2.17.8.1.3.	

Table 2.17-148: SaveFireTargets Information.

2.17.8.1.12 ReadFireTargets

ReadFireTargets reads Fire Targets data from the file specified by the parameter *theFile*. The function call is ReadFireTargets(theFile). Table 2.17-149 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
ft	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
old_ft	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
f	FireTargetDescriptor	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numTargets	int	Standard C type.
i	int	Standard C type.
fireTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h

Calls	
Function	**Where Described**
FSRead	Standard File Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
NewFireTarget	See Section 2.17.4.1.1.
InstallFireTarget	See Section 2.17.4.1.10.
Called By	
Function	**Where Described**
LoadPresets	See Section 2.17.8.1.2.

Table 2.17-149: ReadFireTargets Information.

2.17.8.1.13 WipeFireTable

WipeFireTable wipes out the current Fire Target Table. This routine is called by in order to clear out old Fire Targets data. NewPresets() calls this routine in preparation for the user to set up new Fire Target Table, and LoadPresets() calls this routine in preparation for reading in new canned Fire Target data. The function call is WipeFireTable(). Table 2.17-150 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
ft	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
rem_ft	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
fireTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
UnhookFireTarget	See Section 2.17.4.1.15.	
RemoveScrollTableEntry	See Section 2.22.1.28.1.	
Called By		
Function	**Where Described**	
NewPresets	See Section 2.17.8.1.1.	
LoadPresets	See Section 2.17.8.1.2.	

Table 2.17-150: WipeFireTable Information.

2.17.8.1.14 SaveFPFTargets

SaveFPFTargets saves the Final Protective Fire Targets data to the data file specified by the parameter *theFile*. The function call is SaveFPFTargets(theFile). Table 2.17-151 describes the parameter used, errors returned and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
fpft	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numTargets	int	Standard C type.
i	int	Standard C type.
fpfTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
FSWrite	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
SavePresets	See Section 2.17.8.1.3.	

Table 2.17-151 SaveFPFTargets Information.

2.17.8.1.15 ReadFPFTargets

ReadFPFTargets reads Final Protective Fire Targets data from the file specified by the parameter *theFile*. The function call is ReadFPFTargets(theFile). Table 2.17-152 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
fpft	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
old_fpft	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
f	FPFTargetDescriptor	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numTargets	int	Standard C type.
i	int	Standard C type.
fpfTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h

Calls	
Function	**Where Described**
FSRead	Standard File Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
NewFPFTarget	See Section 2.17.4.2.1.
InstallFPFTarget	See Section 2.17.4.2.9.
Called By	
Function	**Where Described**
LoadPresets	See Section 2.17.8.1.2.

Table 2.17-152: ReadFPFTargets Information.

2.17.8.1.16 WipeFPFTable

WipeFPFTable wipes out the current Final Protective Fire Targets Table. This routine is called in order to clear out old FPFTargets data. NewPresets() calls this routine in preparation for the user to set up new FPFTarget Table, and LoadPresets() calls this routine in preparation for reading in new canned FPFTarget data. The function call is WipeFPFTable(). Table 2.17-153 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
fpft	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
rem_fpft	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
fpfTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
RemoveScrollTableEntry	See Section 2.22.1.28.1.	
Called By		
Function	**Where Described**	
NewPresets	See Section 2.17.8.1.1.	
LoadPresets	See Section 2.17.8.1.2.	

Table 2.17-153: WipeFPFTable Information.

2.17.8.1.17 SaveSchedMissions

SaveSchedMissions saves Scheduled Missions data to the file specified by the parameter *theFile*. The function call is SaveSchedMissions(theFile). Table 2.17-154 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
Schedt	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numMissions	int	Standard C type.
i	int	Standard C type.
targetNum	long	Standard C type.
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
FSWrite	Standard File Manager function for Macintosh.	
ShowCaution	See Section 2.22.1.4.1.	
Called By		
Function	**Where Described**	
SavePresets	See Section 2.17.8.1.3.	

Table 2.17-154: SaveSchedMissions Information.

2.17.8.1.18 ReadSchedMissions

ReadSchedMissions reads Scheduled Missions data from the file specified by the parameter *theFile*. The function call is ReadSchedMissions(theFile). Table 2.17-155 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
theFile	int	Standard C type.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
Schedt	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
old_Schedt	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
f	SchedMissionDescriptor	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
count	long	Standard C type.
numMissions	int	Standard C type.
i	int	Standard C type.
aBadMission	int	Standard C type.
targetNum	long	Standard C type.
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h

Calls	
Function	**Where Described**
FSRead	Standard File Manager function for Macintosh.
ShowCaution	See Section 2.22.1.4.1.
InstallScrollTableEntry	See Section 2.22.1.21.1.
CheckSchedMission	See Section 2.17.8.1.19.
SetSchedMission	See Section 2.17.4.3.16.

Called By	
Function	**Where Described**
LoadPresets	See Section 2.17.8.1.2.

Table 2.17-155 ReadSchedMissions Information.

2.17.8.1.19 CheckSchedMission

CheckSchedMission checks the Scheduled Missions Table specified by *smPtr*. For any missions which were scheduled for a time previous to the current time and were not started, -1 is returned. The function call is CheckSchedMission(smPtr). Table 2.17-156 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
smPtr	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
now	unsigned long	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
-1	int	Mission not ready.

Calls	
Function	**Where Described**
GetDateTime	Standard Operating System Utility function for Macintosh.
Called By	
Function	**Where Described**
ReadSchedMissions	See Section 2.17.8.1.18.

Table 2.17-156: CheckSchedMission Information.

2.17.8.1.20 WipeSchedTable

WipeSchedTable wipes out the current Scheduled Missions Table. This routine is called in order to clear out old Scheduled Missions data. NewPresets() calls this routine in preparation for the user to set up new Scheduled Missions Table, and LoadPresets() calls this routine in preparation for reading in new canned Scheduled Missions data. The function call is WipeSchedTable(). Table 2.17-157 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
resCode	int	Standard C type.
Schedt	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
rem_Schedt	SchedMissionHandle	Development:SIMNET:MCC:FSE:FSEMac.h
astr	array of 128 char	Standard C type.
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
Calls		
Function	**Where Described**	
RemoveScrollTableEntry	See Section 2.22.1.28.1.	
Called By		
Function	**Where Described**	
NewPresets	See Section 2.17.8.1.1.	
LoadPresets	See Section 2.17.8.1.2.	

Table 2.17-157: WipeSchedTable Information.

2.17.8.2 load.c

Development:SIMNET:MCC:FSE:load.c

load.c loads canned data into application data structures, in the standalone, demo version of the FSE application. These functions are only compiled if VERSION is FULDAGAP. Table 2.17-158 describes the variables used in load.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
cannedFireTargets	array of CannedFireTarget	Development:SIMNET:MCC:FSE:load.c
cannedFPFTargets	array of CannedFPFTarget	Development:SIMNET:MCC:FSE:load.c

Table 2.17-158: load.c Variable Information.

2.17.8.2.1 LoadCannedTerrainMap

LoadCannedTerrainMap loads canned information about the exercise terrain. The function call is LoadCannedTerrainMap(). Table 2.17-159 describes the function which calls this function.

Called By	
Function	**Where Described**
main	See Section 2.17.8.3.1.

Table 2.17-159: LoadCannedTerrainMap Information.

2.17.8.2.2 LoadCannedStatus

LoadCannedStatus loads canned mission and battery status. The function call is LoadCannedStatus(). Table 2.17-160 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
battery	unsigned short	Standard C type.
half	unsigned short	Standard C type.
ammo	unsigned short	Standard C type.
map	MapCoordinates	Development:SIMNET:libmac:map.h
Calls		
Function	**Where Described**	
StringToMapCoordinates	See Section 2.22.1.26.1.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-160: LoadCannedStatus Information.

2.17.8.2.3 LoadCannedFireTargets

LoadCannedFireTargets loads precanned target number assignments. The function call is LoadCannedFireTargets(). Table 2.17-161 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ct	pointer to CannedFireTarget	Development:SIMNET:MCC:FSE:load.c
t	FireTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.c
Calls		
Function	**Where Described**	
NewFireTarget	See Section 2.17.4.1.1.	
StringToMapCoordinates	See Section 2.22.1.26.1.	
InstallFireTarget	See Section 2.17.4.1.10.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-161: LoadCannedFireTargets Information.

2.17.8.2.4 LoadCannedFPFTargets

LoadCannedFPFTargets loads precanned FPF targets. The function call is LoadCannedFPFTargets(). Table 2.17-162 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
ct	pointer to CannedFPFTarget	Development:SIMNET:MCC:FSE:load.c
t	FPFTargetHandle	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
NewFPFTarget	See Section 2.17.4.2.1.	
StringToMapCoordinates	See Section 2.22.1.26.1.	
InstallFPFTarget	See Section 2.17.4.2.9.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-162: LoadCannedFPFTargets Information.

2.17.8.3 main.c
Development:SIMNET:MCC:FSE:main.c

main.c contains the CAS application's program entry point and main event loop. Table 2.17-163 describes the variables used in main.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
startupComplete	char	Standard C type.

Table 2.17-163: main.c Variable Information.

2.17.8.3.1 main

main is the program entry point for the FSE console application. The function call is main(). Table 2.17-164 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
myEvent	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
SetUp	See Section 2.17.8.6.1.	
ProcessRequest	See Section 2.17.8.3.6.	
SetUpAppleTalk	See Section 2.22.1.3.1.	
DownloadTerrainMap	See Section 2.22.1.3.4.	
DownloadGunData	See Section 2.17.8.3.2.	
LoadCannedTerrainMap	See Section 2.17.8.2.1.	
LoadCannedStatus	See Section 2.17.8.2.2.	
SetUpMenus	See Section 2.17.8.4.1.	
AddFileMenu	See Section 2.17.8.4.2.	
Install Clock	See Section 2.22.1.6.4.	
SetUpBattery	See Section 2.17.2.1.2.	
SetUpStatusWindow.	See Section 2.17.2.2.1.	
SetUpFireMission	See Section 2.17.1.2.1.	
SetUpFireTable	See Section 2.17.4.1.2.	
SetUpFPFMission	See Section 2.17.1.4.1.	
SetUpFPFTable	See Section 2.17.4.2.2.	
SetUpSchedTable	See Section 2.17.4.3.2.	
LoadCannedFireTargets	See Section 2.17.8.2.3.	
LoadCannedFPFTargets	See Section 2.17.8.2.4.	
ShowStatusWindow	See Section 2.17.2.2.2.	
MainEventLoop	See Section 2.17.8.3.3.	

Table 2.17-164: main Information.

2.17.8.3.2 DownloadGunData

DownloadGunData downloads information about the fire units. This function is compiled only if VERSION is APPLETALK. The function call is DownloadGunData(). Table 2.17-165 describes the internal variable used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
myEvent	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.	
NetworkEventHandler	See Section 2.22.1.3.5.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-165: DownloadGunData Information.

2.17.8.3.3 MainEventLoop

MainEventLoop sits in a loop forever fielding incoming events to the FSE console. The function call is MainEventLoop(). Table 2.17-166 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
distanceCheat	extern short	Standard C type.
myEvent	EventRecord	Development:THINK C: Mac #includes:EventMgr.h
Calls		
Function	**Where Described**	
CheckTimers	See Section 2.17.5.1.1.	
UpdateClock	See Section 2.22.1.6.2.	
SystemTask	Standard Desk Manager function for Macintosh.	
GetNextEvent	Standard Toolbox Event Manager function for Macintosh.	
AddFileMenu	See Section 2.17.8.4.2.	
ShowDisplace	See Section 2.17.3.1.1.	
ATPCloseSocket	Standard Appletalk Manager function for Macintosh.	
ExitToShell	Standard Segment Loader function for Macintosh.	
SaveTerrainMap	See Section 2.17.8.1.7.	
ShowVersions	See Section 2.22.1.44.1.	
AdjustScreenState	See Section 2.17.8.3.4.	
WindowEvent	See Section 2.22.1.46.2.	
NetworkEventHandler	See Section 2.22.1.3.5.	

Called By	
Function	**Where Described**
main	See Section 2.17.8.3.1.

Table 2.17-166: MainEventLoop Information.

2.17.8.3.4 AdjustScreenState

AdjustScreenState updates the knowledge about the state of a screen on a window, *w*, becoming active. The function call is AdjustScreenState(w). Table 2.17-167 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
w	register WindowPtr	Development:THINK C: Mac #includes:WindowMgr.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
c	register long	Standard C type.
Calls		
Function	**Where Described**	
EnableMenus	See Section 2.17.8.4.4.	
Called By		
Function	**Where Described**	
MainEventLoop	See Section 2.17.8.3.3.	

Table 2.17-167: AdjustScreenState Information.

2.17.8.3.5 ShowHelp

ShowHelp puts up a help screen. Dialogs that are part of the FSE application have help topics associated with them that are numbered the same as the dialogs' DLOG resources. The function call is ShowHelp(). Table 2.17-168 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
topic	int	Standard C type.
c	register long	Standard C type.
Calls		
Function	**Where Described**	
GetWRefCon	Standard Window Manager function for Macintosh	
showhelp	See Section 2.22.1.19.4.	

Called By	
Function	**Where Described**
MenuCommand	See Section 2.17.8.4.5.

Table 2.17-168: ShowHelp Information.

2.17.8.3.6 ProcessRequest

ProcessRequest processes a request received from the MCC host. This function is compiled only when VERSION is APPLETALK. The function call is ProcessRequest(hdl). Table 2.17-169 describes the parameters used, and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
hdl	register Handle	Development:THINK C: Mac #includes:macTypes.h
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
p	register Ptr	Development:THINK C: Mac #includes:MacTypes.h
b	register pointer to Battery	Development:SIMNET:MCC: FSE:FSEMac.h
i	short	Standard C type.
errCode	int	Standard C type.
Calls		
Function	**Where Described**	
SetDateTime	Standard Operating Sytem Utility function for Macintosh.	
WithdrawBattery	See Section 2.17.5.2.3.	
PointToMapCoordinates	See Section 2.22.1.26.2.	
ThrowDisplace	See Section 2.17.3.1.2.	
UpdateBatteryDisplay	See Section 2.17.2.1.7.	
UpdateAmmoDisplay	See Section 2.17.2.1.8.	
ATPResponse	Standard Appletalk Manager function for Macintosh.	
Restart	Standard Operating System Utility function for Macintosh.	
RemoteUnitDispatched	See Section 2.17.3.2.3.	
RemoteUnitArrived	See Section 2.17.3.2.4.	
Called By		
Function	**Where Described**	
main	See Section 2.17.8.3.1.	

Table 2.17-169: ProcessRequest Information.

2.17.8.4 menu.c

Development:SIMNET:MCC:FSE:menu.c

menu.c implements the menus presented by the FSE application. Table 2.17-170 describes the variables used in menu.c.

Variables		
Variable	**Type**	**Where Typedef Declared**
menu	array of numberMenues struct menus	Development:SIMNET:MCC: FSE:menu.c

Table 2.17-170: menu.c Variable Information.

2.17.8.4.1 SetUpMenus

SetUpMenus creates a menu bar. The function call is SetUpMenus(). Table 2.17-171 describes the internal variable used and the functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
i	register short	Standard C type.
Calls		
Function	**Where Described**	
InitMenus	Standard Menu Manager function for Macintosh.	
InsertMenu	Standard Menu Manager function for Macintosh.	
DrawMenuBar	Standard Menu Manager function for Macintosh.	
DisableItem	Standard Menu Manager function for Macintosh.	
SetMenuHandler	See Section 2.22.1.46.1.	
Called By		
Function	**Where Described**	
main	*See Section 2.17.8.3.1.*	

Table 2.17-171: SetUpMenus Information.

2.17.8.4.2 AddMenuFile

AddMenuFile inserts a menu file into the menu bar. The function call is AddMenuFile(). Table 2.17-172 describes the functions called using this function.

Calls	
Function	**Where Described**
InsertMenu	Standard Menu Manager function for Macintosh.
GetMenu	Standard Menu Manager function for Macintosh.
DrawMenuBar	Standard Menu Manager function for Macintosh.
Called By	
Function	**Where Described**
main	See Section 2.17.8.3.1.
MainEventLoop	See Section 2.17.8.3.3.

Table 2.17-172: AddFileMenu Information.

2.17.8.4.3 FutureMissionTest

FutureMissionTest is used to determine if all selected scheduled missions are future missions. The function call is FutureMissionTest(defn, selection). Table 2.17-173 describes the parameters used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
selection	ScrollTableEntryHandle	Development:SIMNET:libmac: scroll.h

Table 2.17-173: FutureMissionTest Information.

2.17.8.4.4 EnableMenus

EnableMenus enables and disables menu items according to what is currently on the screen. The function call is EnableMenus(). Table 2.17-174 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
fireTableField	extern ScrollTableFieldDefn	Development:SIMNEt:libmac: scroll.h
fpfTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
i	register short	Standard C type.
defn	pointer to ScrollTableFieldDefn	Development:SIMNET:libmac: scroll.h
Calls		
Function	**Where Described**	
DisableItem	Standard Menu Manager function for Macintosh.	
EnableItem	Standard Menu Manager function for Macintosh.	
MapScrollTableSelected	See Section 2.22.1.33.1.	
MortarBattery	Macro defined in FSEMac.h. See Section 2.17.7.2.	
Called By		
Function	**Where Described**	
MenuCommand	See Section 2.17.8.4.5.	
FireTableSelect	See Section 2.17.4.1.5.	
FPFTableSelect	See Section 2.17.4.2.4.	
SchedTableSelect	See Section 2.17.4.3.4.	
SetSchedMission	See Section 2.17.4.3.16.	
AdjustScreenState	See Section 2.17.8.3.4.	

Table 2.17-174: EnableMenus Information.

2.17.8.4.5 MenuCommand

MenuCommand performs a command chosen from a menu. The function call is MenuCommand(mResult). Table 2.17-175 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
mResult	unsigned long	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
fireTableDialog	extern pointer to DialogState	Development:SIMNET:libmac:dialog.h
fpfTableDialog	extern pointer to DialogState	Development:SIMNET:libmac:dialog.h
schedTableDialog	extern pointer to DialogState	Development:SIMNET:libmac:dialog.h
fireTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
fpfTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
schedTableField	extern ScrollTableFieldDefn	Development:SIMNET:libmac:scroll.h
theMenu	int	Standard C type.
theItem	int	Standard C type.

Calls	
Function	**Where Described**
ShowHelp	See Section 2.17.8.3.5.
ShowStatusWindow	See Section 2.17.2.2.2.
ShowFireTable	See Section 2.17.4.1.4.
ShowFPFTable	See Section 2.17.4.2.3.
ShowSchedTable	See Section 2.17.4.3.3.
UndoFireTarget	See Section 2.17.4.1.14.
UndoFPFTarget	See Section 2.17.4.2.13.
UndoSchedMission	See Section 2.17.4.3.14.
UndoDisplace	See Section 2.17.3.1.6.
ShowFireTarget	See Section 2.17.4.1.11.
NewFireTarget	See Section 2.17.4.1.1.
ShowFPFTarget	See Section 2.17.4.2.10.
NewFPFTarget	See Section 2.17.4.2.1.
ShowSchedMission	See Section 2.17.4.3.11.
NewSchedMission	See Section 2.17.4.3.1.
MapScrollTableSelected	See Section 2.22.1.33.1.
EnableMenus	See Section 2.17.8.4.4.
ShowNewFireMission	See Section 2.17.1.5.1.
ShowNewFPFMission	See Section 2.17.1.4.3.
ShowDisplace	See Section 2.17.3.1.1.
SavePresets	See Section 2.17.8.1.3.
LoadPresets	See Section 2.17.8.1.2.
NewPresets	See Section 2.17.8.1.1.
HiliteMenu	Standard Menu Manager function for Macintosh.

Table 2.17-175: MenuCommand Information.

2.17.8.5 mission.c

Development:SIMNET:MCC:FSE:mission.c

mission.c contains routines for creating new fire mission descriptors and updating the display of a mission's status.

2.17.8.5.1 NewMission

NewMission finds an unused mission descriptor and returns it in *m*. The function call is NewMission(type). Table 2.17-176 describes the parameter used by this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
type	int	Standard C type.
Internal Variables		
Variable	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
i	short	Standard C type.
j	short	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
0	pointer to Mission	Maximum numbers of missions already in use.
m	pointer to Mission	Unused mission descriptor.
Called By		
Function	**Where Described**	
ShowNewFPFMission	See Section 2.17.1.4.3.	
ShowNewFireMission	See Section 2.17.1.5.1.	

Table 2.17-176: NewMission Information.

2.17.8.5.2 EndMission

EndMission frees up a mission descriptor, *m*. The function call is EndMission(m). Table 2.17-177 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
RedrawMissionIcon	See Section 2.17.2.2.5.	
Called By		
Function	**Where Described**	
EndFireMission	See Section 2.17.1.2.10.	
FPFEvent	See Section 2.17.1.4.6.	
NewMissionEvent	See Section 2.17.1.5.2.	
StartSchedMission	See Section 2.17.4.3.15.	

Table 2.17-177: EndMission Information.

2.17.8.5.3 OpenMission

OpenMission opens an icon to display the status of a mission, *m*. The function call is OpenMission(m). Table 2.17-178 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
OpenFireMission	See Section 2.17.1.2.3.	
OpenFPFMission	See Section 2.17.1.4.4.	

Called By	
Function	**Where Described**
StatusEventHandler	See Section 2.17.2.2.4.

Table 2.17-178: OpenMission Information.

2.17.8.5.4 ToggleBattery

ToggleBattery toggles the allocation of a battery, described by *b* and *half*, to a current mission. The function call is ToggleBattery(b, half). Table 2.17-179 describes the parameters used, and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
b	int	Standard C type.
half	int	Standard C type.
Calls		
Function	**Where Described**	
StartMissionBattery	See Section 2.17.5.1.6.	
EndMissionBattery	See Section 2.17.5.1.7.	
Called By		
Function	**Where Described**	
FireEvent	See Section 2.17.1.2.5.	
FPFEvent	See Section 2.17.1.4.6.	

Table 2.17-179: ToggleBattery Information.

2.17.8.5.5 UpdateMissionDisplay

UpdateMissionDisplay updates the screen for a mission, *m*, whose status may have changed. The function call is UpdateMissionDisplay(m). Table 2.17-180 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
RedrawMissionIcon	See Section 2.17.2.2.5.	
UpdateBatterySelection	See Section 2.17.5.2.2.	
UpdateFireDialog	See Section 2.17.1.2.7.	
UpdateFPFDialog	See Section 2.17.1.4.9.	

Called By	
Function	**Where Described**
AdjustEvent	See Section 2.17.1.1.2.
FFEEvent	See Section 2.17.1.3.3.
CommenceFFE	See Section 2.17.1.3.4.
CancelAtCommand	See Section 2.17.1.3.5.
CheckFiring	See Section 2.17.1.3.6.
FPFEvent	See Section 2.17.1.4.6.
StartSchedMission	See Section 2.17.4.3.15.
FireVolley	See Section 2.17.5.1.2.
WithdrawBattery	See Section 2.17.5.2.1.

Table 2.17-180: UpdateMissionDisplay Information.

2.17.8.5.6 UpdateMissionState

UpdateMissionState updates the display of the mission state, *m*. The function call is UpdateMissionState(m). Table 2.17-181 describes the parameter used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
m	register pointer to Mission	Development:SIMNET:MCC:FSE:FSEMac.h
Calls		
Function	**Where Described**	
RedrawMissionIcon	See Section 2.17.2.2.5.	
UpdateFireState	See Section 2.17.1.2.8.	
UpdateFPFState		
Called By		
Function	**Where Described**	
CheckTimers	See Section 2.17.5.1.1.	

Table 2.17-181: UpdateMissionState Information.

2.17.8.5.7 UpdateObserver

UpdateObserver updates the display of an Observer's Call Sign everywhere. This function is called by the libmac dialog code as a field validation function associated with the Observer field in the mission status dialog. The function call is UpdateObserver(dialog, fp, str). Table 2.17-182 describes the parameters used and functions called using this function.

Parameters		
Parameter	**Type**	**Where Typedef Declared**
dialog	pointer to DialogState	Development:SIMNET:libmac:dialog.h
fp	pointer to TextFieldDefn	Development:SIMNET:libmac:dialog.h
str	pointer to char	Standard C type.

Return Values		
Return Value	**Type**	**Meaning**
1	int	Always returned.

Calls	
Function	**Where Described**
RedrawMissionIcon	See Section 2.17.2.2.5.
ApplyToBatteries	See Section 2.17.5.1.5.

Table 2.17-182: UpdateObserver Information.

2.17.8.6 setup.c

Development:SIMNET:MCC:FSE:setup.c

setup.c initializes the FSE application at start-up.

2.17.8.6.1 SetUp

SetUp initializes the SIMNET FireSupportElement application. The toolbox is intialized. The SIMNET Resources and FSE Pictures files are opened. The Help Manager is initialized. Storage is allocated for pop-up dialogs. The table of missions and table of batteries are initialized. The function call is SetUp(). Table 2.17-183 describes the internal variables used and functions called using this function.

Internal Variables		
Variable	**Type**	**Where Typedef Declared**
errCode	int	Standard C type.
finalTicks	long	Standard C type.
i	register short	Standard C type.

Calls	
Function	**Where Described**
InitToolbox	See Section 2.22.1.20.1.
ZoomInit	See Section 2.22.1.47.1.
OpenResFile	Standard Resource Manager function for Macintosh.
MenuBarTitle	See Section 2.22.1.43.1.
Delay	Standard Operating System Utility function for Macintosh.
ExitToShell	Standard Segment Loader function for Macintosh.
DeepShit	See Section 2.22.1.8.2.
helpinit	See Section 2.22.1.19.1.
SystemFailure	See Section 2.22.1.8.1.
NewPtr	Standard Memory Manager function for Macintosh.

Called By	
Function	**Where Described**
main	See Section 2.17.8.3.1.

Table 2.17-183: SetUp Information.

2.18 The CAS (Close Air Support) Console

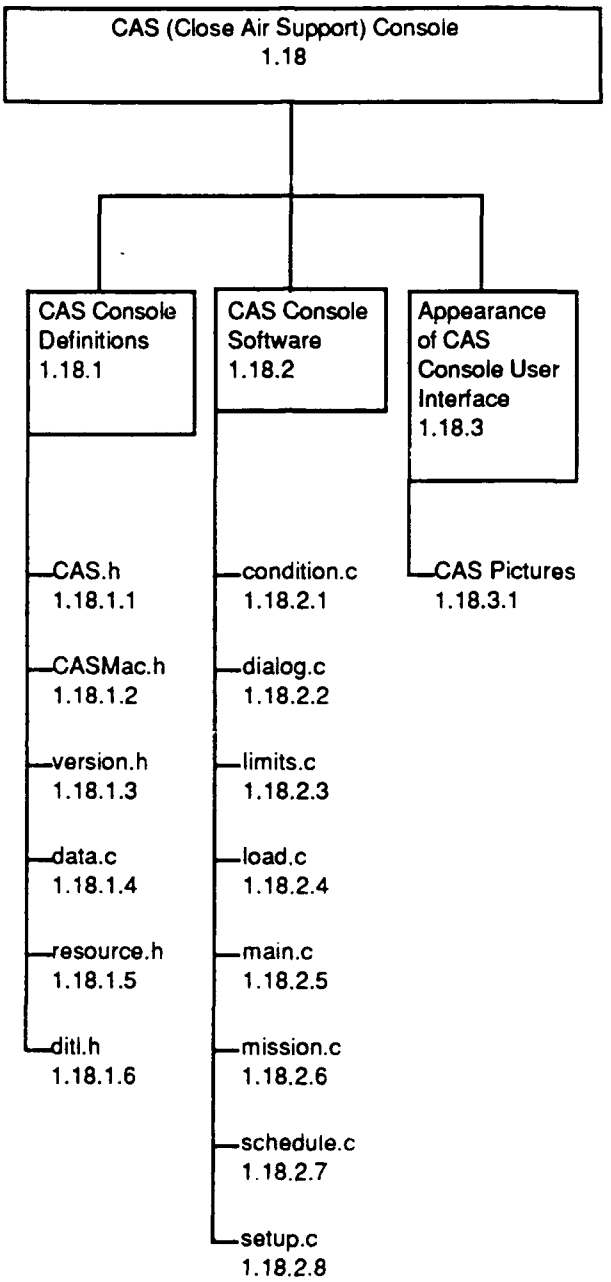


Figure 2.18-1: CAS (Close Air Support) Console Structure.