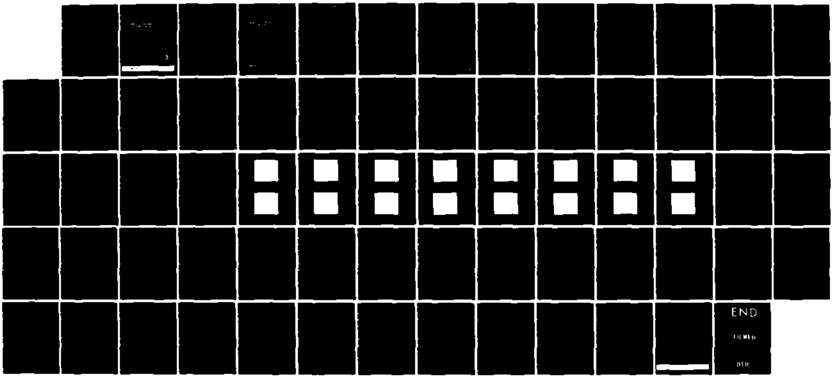
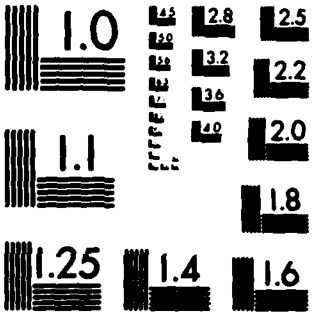
AD-A150 569   TWIRL: TACTICAL WARFARE IN THE ROSS LANGUAGE(U) RAND   1/1
CORP SANTA MONICA CA P KLAHR ET AL. OCT 84
RAND/R-3158-AF F49620-82-C-0018

UNCLASSIFIED   F/G 15/7   NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

2

R-3158-AF

TWIRL: Tactical Warfare in the ROSS Language

Philip Klahr, John W. Ellis, Jr., William Giarla,
Sanjai Narain, Edison M. Cesar, Jr., Scott R. Turner

October 1984

"Criginal contains color plates: All DTIC reproductions will be in black and white"

Rand

The research reported here was sponsored by the Directorate of Operational Requirements, Deputy Chief of Staff/Research, Development, and Acquisition, Hq USAF, under Contract F49620-82-C-0018. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

Library of Congress Cataloging in Publication Data
Main entry under title:

TWIRL : tactical warfare in the ROSS language.

"A Project Air Force report prepared for the United States Air Force."
"R-3158-AF."
Bibliography: p.
1. Stream crossing, Military—Data processing.
2. Tactics—Data processing. 3. ROSS (Computer program language) I. Klahr, Philip. II. Rand Corporation.
III. United States. Air Force.
U206.T87 1984 355.4'22 84-20663
ISBN 0-8330-0601-0

The Rand Publications Series: The Report is the principal publication documenting and transmitting Rand's major research findings and final research results. The Rand Note reports other outputs of sponsored research for general distribution. Publications of The Rand Corporation do not necessarily reflect the opinions or policies of the sponsors of Rand research.

Published by The Rand Corporation

R-3158-AF

TWIRL: Tactical Warfare in the ROSS Language

Philip Klahr, John W. Ellis, Jr., William Giarla,
Sanjai Narain, Edison M. Cesar, Jr., Scott R. Turner

October 1984

A Project AIR FORCE Report
prepared for the
United States Air Force

Rand
1700 MAIN STREET
P.O. BOX 2138
SANTA MONICA, CA 90406-2138

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
R-3158-AF

2. GOVT ACCESSION NO.
AD-A150569

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
TWIRL: Tactical Warfare in the ROSS Language

5. TYPE OF REPORT & PERIOD COVERED
Interim

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Philip Klahr, John W. Ellis, Jr., William Giarla, Sanjai Narain, Edison Cesar, Jr., Scott R. Turner

8. CONTRACT OR GRANT NUMBER(s)
F49620-82-C-0018

9. PERFORMING ORGANIZATION NAME AND ADDRESS
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Requirements, Programs and Studies Group (AF/RDQM)
Ofc, DSC/R&D And Acquisition
Hq USAF, Washington, DC 20330

12. REPORT DATE
October 1984

13. NUMBER OF PAGES
49

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
No Restrictions

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Programming Languages	Electronic Warfare
Computerized Simulation	Tactical Warfare
Models	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

see reverse side

This report describes TWIRL, a simulation of a primarily ground combat engagement between two opposing military forces. It was developed to further experiment with the ROSS language, an object-oriented simulation language that was successfully used to develop the SWIRL air battle simulation, and to develop a prototype simulation that could be used to explore issues in electronic combat. The authors describe the objects that comprise TWIRL and provide extensive examples of object behaviors to explain and illustrate the process of building a simulation in ROSS.

PREFACE

As part of a concept-development effort within the Force Employment Program of Project AIR FORCE, Rand has been investigating techniques to improve computer technology for military combat simulations. This work has focused on developing a research environment that aids users in designing, building, understanding, and modifying battle simulations in order to analyze and evaluate various outcomes. Results of this work include the development of an English-like object-oriented simulation language called ROSS, the design and implementation of a strategic air penetration simulation built in ROSS called SWIRL, and the design of a parallel processing computer architecture for distributing simulations to significantly enhance their performance. These systems are reported in the following Rand publications:

ROSS: An Object-Oriented Language for Constructing Simulations, by D. McArthur, P. Klahr, and S. Narain, R-3160-AF (forthcoming).

The ROSS Language Manual, by D. McArthur and P. Klahr, N-1854-AF, September 1982.

SWIRL: Simulating Warfare in the ROSS Language, by P. Klahr, D. McArthur, S. Narain, and E. Best, N-1885-AF, September 1982.

Fast Concurrent Simulation Using the Time Warp Mechanism, Part I: Local Control, by D. Jefferson and H. Sowizral, N-1906-AF, December 1982.

This report describes another ROSS simulation called TWIRL, which simulates a military engagement involving primarily ground combat between two opposing tactical forces. TWIRL represents a prototype of a simulation system that could be used to experiment with alternative strategies and tactics in electronic combat and physically destructive warfare. The TWIRL development has been sponsored jointly by Project AIR FORCE and The Rand Corporation.

SUMMARY

This report describes TWIRL, a simulation of a primarily ground combat engagement between two opposing military forces. It was developed to further experiment with the ROSS language, an object-oriented simulation language that was successfully used to develop the SWIRL air battle simulation (Klahr et al., 1982a, 1982b), and to develop a prototype simulation that could be used to explore issues in electronic combat.

The domain selected for TWIRL is a hasty river crossing operation (executed from the march). Such a military maneuver employs a combined arms team, with complex interactions and critical timing requirements among the various arms and activities. These characteristics make it a suitable vehicle within which to examine and illustrate the potential disruptive effects of electronic combat.

We describe the objects that comprise TWIRL and provide extensive examples of object behaviors to explain and illustrate the process of building a simulation in ROSS. Many examples of ROSS/TWIRL code are included.

A major component of TWIRL is a color graphics facility which dynamically displays simulations as they are running. This capability is highlighted by several color plates taken from TWIRL runs. These color plates not only illustrate the TWIRL simulation, they also demonstrate the potential payoff of graphics (and other human-computer interactive tools) for modeling and understanding complex problems.

ACKNOWLEDGMENTS

TWIRL embodies considerable knowledge about tactical ground combat and electronic warfare. We wish to thank Lt. Gen. (Ret.) John Cushman, U.S. Army, and John Callahan and Kenneth Smith of Defense Systems Corporation for sharing their views and knowledge in these areas. We have also benefited greatly from discussions with Rand colleagues David McArthur, Milton G. Weiner, and Willard Naslund, and with Dennis Bennett of TRW, who provided the much-needed initial impetus and motivation for this work.

CONTENTS

FIGURES

COLOR PLATES
(Following p. 28)

I. OVERVIEW AND GOALS

During the past few years, we have been exploring methods for improving simulation technology. These studies have applied and extended recent advances in artificial intelligence, expert systems, computer networking, parallel processing, and graphics to the area of simulation. We have sought to create an environment, or simulation laboratory, where credible simulations can be easily built, easily understood and modified, and run at speeds acceptable for human use.

This research has produced ROSS (McArthur and Klahr, 1982; McArthur et al., 1984), an object-oriented message-passing language ideally suited for developing simulations; the SWIRL simulation (Klahr et al., 1982a, 1982b), a strategic air battle simulation written in ROSS; and the Time Warp mechanism (Jefferson and Sowizral, 1982), an approach to using parallel processing to significantly speed up simulation runs. This report describes our most recent development, TWIRL (Tactical Warfare in the ROSS Language), a ground combat simulation.

The TWIRL development was undertaken primarily to

- Further explore and experiment with the ROSS environment to better determine its strengths and weaknesses for simulating military battles.
- Determine how to represent and visually depict electronic systems operating on the battlefield.
- Design and implement a prototype simulation that could serve as a tool for exploring various strategies and tactics in electronic combat.
- Develop a (prototype) capability that allows a human to take the role of one of the objects involved in a battle. For use in analysis or training, this facility would enable a human to play an active, decisionmaking role during a simulation run.

Our experience with the development of the SWIRL air battle simulation convinced us that the ROSS language is fully capable of representing and simulating military battles, primarily because of its object-oriented framework for expressing the simulation domain. Briefly, one formulates a set of *objects* representing classes (e.g., radars, aircraft, missiles, etc.) or individual members of those classes (e.g., radar14), and specifies their attributes and behaviors. Objects are organized in a hierarchy of class-subclass links (e.g., GCI radars might be a subclass of ground-based radars, which in turn might be a subclass of radars). This hierarchy allows an object to "inherit" automatically the attributes and behaviors of the classes to which it belongs. Thus, for example, ground-based radars, being a subclass of radars, would inherit knowledge associated with the radar class (the general class of all radars). The organization of knowledge around objects facilitates modularity, modifiability, and maintenance of the simulation (Klahr and Faught, 1980).

Within ROSS, objects communicate with one another by sending *messages*. These messages are patterns or templates that typify the real-world messages that objects would normally send to one another in a battle (e.g., airborne surveillance radar to operations center: "Enemy aircraft detected 120 km west of Berlin"). When receiving a particular message, an object searches through its behaviors to match the message and then takes the corresponding specified action (e.g., operations center launches defensive fighters). The concepts of *objects*, *messages*, and *messages triggering actions* form the basis of the ROSS language.

Following the success of using ROSS for the SWIRL simulation, we wished to expand its application to other environments, in particular, a tactical, land-based combat environment. Because of the growing interest in electronic combat (EC), both at Rand and in the military, we decided to focus on representing communications and exploring various strategies for interfering with communications during combat. For our prototype simulation, we chose to model destruction (by artillery fire), disruption (by jamming), and some limited air operations.

Section II of this report describes the TWIRL domain. Sections III through V describe the TWIRL implementation, the various design decisions that were made, and how they evolved into ROSS code. Section V also provides examples of how the TWIRL simulation could be used as a training facility. In particular, we have developed an initial capability to allow a human to play one of the simulation objects. The human player receives and sends the messages the corresponding automated object would typically receive and send in a simulation. This *human-in-the-loop* facility adds another dimension to the interactivity of ROSS. A human could also use this facility to force certain decisions to be made and thereby direct the simulation to whatever experimental situation is desired. Thus, this capability can contribute to analysis, training, and the development of tactics.

The reader should keep in mind that the particular operation we have simulated is a fictitious one. Our primary concern is with methodology—we are building tools and exploring methods to help humans better depict and understand complex phenomena. Therefore, some of the particular objects and behaviors we have chosen to represent are simplistic and some may even be incorrect.[1] The main point we are attempting to make is that the ROSS environment is rich enough to allow one to easily understand and modify the objects and behaviors to create whatever simulation model is desired.

A primary objective of this report is to indicate what it is like to build a battle simulation in ROSS. We often specify various objects and behaviors in ROSS code; however, a knowledge of ROSS is not necessary to fully understand the material.

[1] Some of the behaviors presented here have been even further simplified for exposition purposes. Also, many objects and behaviors that normally figure in ground combat have been omitted to reduce the scope and complexity of our prototype effort.

II. GENERAL DESCRIPTION OF TWIRL

TWIRL simulates ground combat between two opposing forces (hereafter referred to as Red and Blue). We view TWIRL as a prototype of a design tool for military strategists working in command, control, and communications countermeasures (C^3CM), electronic warfare (EW), and electronic combat (EC). TWIRL consists of a set of ROSS-defined objects representing the players in the simulation, along with their specified attributes and possible behaviors. Given an initial configuration of Red and Blue forces, TWIRL simulates their movements and interactions over time.

A highly useful component of TWIRL is its color graphics facility. The use of color greatly aids in differentiating between different types of objects, e.g., Red and Blue; in contrasting movable objects with the physical setting, i.e., the terrain (mountains, rivers, streams, roads); and in locating the various activities performed by the objects, i.e., bombing, artillery firing, communicating, jamming, and direction finding.

In a TWIRL simulation, the user receives both textual information about the simulated events and visual information that depicts the battle as it proceeds over time. In addition to the usual map display of unit identities and locations, TWIRL incorporates visual indications of communications traffic (identifying originator and recipient(s)) and selected EW activities (direction finding and jamming). In a sense, the simulation presents an animated movie of the participating units, augmented by displays of the activities in the electromagnetic spectrum.

Since the graphics interface directly with the simulation, the user sees the simulation as it is produced. Also, because ROSS is highly interactive, the user may stop the simulation and graphics at any time, zoom or pan, examine or change various attributes or behaviors, etc. Although the graphics must be viewed in a live demonstration to fully appreciate their scope and utility, we have included several color plates of TWIRL's graphical output to give at least some indication of this capability.

PROBLEM DOMAIN: HASTY RIVER CROSSING OPERATION

The domain we chose for TWIRL is a hasty river crossing operation by Red forces (i.e., one executed directly from the approach march without delay for assembling massive assault forces and fire support). Such a military maneuver employs a combined arms team (infantry, artillery, armor, electronic combat, etc.), with complex interactions and critical timing requirements among the various arms and activities. These characteristics make it a suitable vehicle for examining and illustrating the potentially disruptive effects of EC. In addition, descriptions of typical Red river crossing operations that include doctrine, timing, force composition, tactics, procedures, and norms can be found in readily available sources (Department of the Army, 1977, 1978; Reznichenko, 1966; Sidorenko, 1970).

In our scenario, Red forces are on the offensive, nearing a sizable river obstacle. As the scenario opens, both the Red division choosen to force the crossing and the likely defending Blue mechanized regiment are in assembly areas some distance from the river. Red's plan calls for two regiments, moving forward abreast, to mount the assault directly from the approach march in an attempt to secure the crossing before defending Blue forces can establish a deliberate defense of the river line. Once in its hasty defense posture, Blue has two main methods of interfering with Red's planned operation: through artillery fire, or by communications jamming (it could also use some combination of the two).[1]

Each Red unit is assigned a movement schedule designed to fit its contribution to the division's overall advance and assault crossing of the river. These schedules (called *track plans*) would be carried out as long as no Blue-induced obstacle or disruptive actions interfere. The assault units' track plans call for the crossing to begin at "H-hour." Thus, the simulation begins at H−7:00 (H minus 7 hours), when the vanguards of the lead regiments move out of the divisional assembly area about 70 km from the river. The Red regiments and supporting division artillery units proceed according to their track plans. Along the way, at preplanned halts, maneuver and artillery units deploy into smaller elements (each with its own track plan). This process continues until all Red units reach their assigned assault positions. There they halt, deploy into combat formations, and await the assault order from their parent regiments.[2] Blue, of course, will seek to disrupt Red's planned advance. Appendix A describes the planned deployment and movement of all Red units included in the TWIRL simulation.

As the Red attack units continue moving forward along the road network, Blue aerial reconnaissance discovers them and communicates this to the Blue ground force commander. He, in turn, orders the defending Blue mechanized regiment to move out of its assembly area and take up defensive positions along the river opposing the expected Red attack axes. In our simulation, each Blue unit also has a predefined track plan of movements. However, for the purpose of this simulation, and unlike Red, no obstacles are permitted to interfere with Blue unit movements to their assigned defensive positions. This insures that a hasty defense is established before Red units begin their assault crossing, thereby providing the necessary context within which to simulate the disruptive effects of jamming and/or artillery fire on the Red timetable. Appendix A also describes the movement and deployment of all Blue units.

As each Red unit proceeds on its track plan, it communicates to its parent regiment whenever it begins or completes a phase of its deployment to its assault positions. No attempt has been made to faithfully replicate the volume and entire range of communications traffic that might accompany such operations. These two types of messages (announcements of the start of road movement and of planned stopovers) were deemed sufficient to allow simulation of Blue ELINT (electronics intelligence), COMINT (communications intelligence), SIGINT (signals intelligence), and target location and generation activities.

Once in their defensive positions and alerted, Blue's direction finders attempt to locate the positions of enemy combat units. Whenever a direction finder learns that a unit has stopped, it notifies the Blue regimental commander of a potential target at a particular position. The Blue commander must then decide, given its jamming and artillery assets, what to do about the target. Jamming blocks a unit from receiving messages on one or more channels, causing a delay in that unit's reception of messages (we assume that when the recipient of a message is jammed, the sender knows it and retransmits the message on another channel or dispatches it by other, nonelectronic means). Artillery fire causes attrition and movement delay when units are fired upon and hit. Red can similarly jam and fire against Blue's units.

[1] Blue offensive air support operations are simulated in only very rudimentary form, primarily to illustrate feasibility and the scope and volume of the additional actors and behaviors needed to couple air activity to the ground combat. To date, only the results of interdiction attacks on bridge and road junction targets have been included in our computations (see Section IV). Although allowed for in the code, Red air operations are not part of the present TWIRL simulation.

[2] To simplify the simulation and still incorporate the description and effects of vital command and control communications, we include only objects at regiment level and below. In practice, the assault order would more likely be the responsibility of the division. In TWIRL, division-level objects play no active role in the simulation.

Blue can initiate air reconnaissance to detect Red's position and movement, and can initiate air attacks on various road junctions and bridges along Red's route to slow the advance. (Section IV discusses how TWIRL could be extended to allow Blue air to engage Red ground or air forces.)

Currently, the major measure of effectiveness of Blue's various jamming, artillery, and air support strategies in TWIRL is the time delay involved in Red's planned assault.[3] If Red were to proceed unimpeded, the Red assault units would begin to cross the river at H-hour. Blue's attempts to stop Red are expected to result in delays to Red's advance, providing a measure of how well Blue has done.[4] A graphic display of the disruption to individual Red unit plans (contributing to the overall delay in the scheduled assault time) would assist greatly in judging the results of different Blue strategies. We have provided simultaneous presentation of an unimpeded and a delayed assault by using two regiments assaulting abreast. One is programmed to be immune to Blue actions directed at its subordinate units so that all proceed according to their scheduled track plans. The two regiments have identical orders of battle, and both are scheduled to begin the river crossing at H-hour. Consequently, the graphic display, at any given time in the simulation, will illustrate the cumulative deviation (from schedule) of any unit that has been affected by Blue's defensive efforts.

Even with this brief description, one can begin to see how such a simulation can be used to examine various situations and problems. Some hypothetical questions a military analyst might address include:

- What units and actions are affected if specific Red communications are disrupted?
- Are any of Red's communication networks critical?
- What happens when Blue jams a crucial node in a network?
- How effective are redundant communications?
- What effects result from Red jamming Blue's communications?
- How can Blue best disrupt a Red river crossing operation?

THE PLAYEFS

TWIRL contains approximately 50 objects that represent classes (or subclasses) and approximately 90 objects that represent actual instances of those classes (i.e., particular Red and Blue units). The number of classes represented may appear high relative to the number of instances. However, many of the class objects serve as intermediate objects in the object hierarchy and do not have instances as direct descendants. (For example, MOVING-OBJECT contains the subclasses RED-UNIT and BLUE-UNIT, which, in turn, eventually spawn particular artillery units, regiments, etc.) Also, most behaviors are associated with classes rather than instances. For our purposes, we are more concerned with the ability to represent various types of military units than we are with the ability to represent large quantities of any particular type of unit. Once we define a division artillery unit, for example, it is of little concern whether we have 4 or 40 of them. Essentially, they all exhibit similar behaviors (in our simulation), and only their particular attributes (e.g., position, strength) may vary.[5]

[3] In addition, provision has been made in the TWIRL code to calculate damage to units under artillery fire and to record their current strength. Thus, the force ratio at the point of contact could be calculated as another measure of merit or for use to control movement once combat was joined.

[4] Section VI contains a brief description and analysis of the events resulting from an exemplary Blue disruption effort and compares those results with the Red scheduled attack plan as given in Appendix A.

[5] Certainly other issues arise when one is working with large numbers (perhaps thousands) of objects, particularly issues of computer memory and speed. Some of these are addressed in Jefferson and Sowizral (1982). Our concern here centers on how best to represent domain knowledge.

The main objects that comprise the TWIRL domain are listed below. The TWIRL name of each object is given, followed by a brief description of what the object represents. More detail on many of these objects will be presented in subsequent sections. Most of the objects correspond to real-world entities, but some are auxiliary objects (they do not have real-world correlates, but they function as full objects, in the ROSS sense, to perform certain computations, e.g., the MATHEMATICIAN and the PHYSICIST (Klahr et al., 1982b)). Objects are organized hierarchically and may belong to many classes simultaneously, thereby giving rise to multiple inheritance hierarchies.

General, high-level objects include the following:

SIMULATOR. The top-level object in TWIRL. All other TWIRL objects are descendants of SIMULATOR (i.e., its subclasses include MOVING-OBJECT, which has subclasses RED-UNIT and BLUE-UNIT, and so on).

MOVING-OBJECT. Contains behaviors used for movement, e.g., stopping, starting, turning, waiting, etc. All Red and Blue units are moving objects (i.e., the objects RED-UNIT and BLUE-UNIT are subclasses of MOVING-OBJECT).

COMMUNICATING-OBJECT. Represents the class of TWIRL objects that can communicate (in the real-world sense, rather than in the ROSS sense) with other objects in the simulation. It contains all the behaviors objects use to send messages over the various communication channels defined in TWIRL. All Red and Blue units are *communicating-objects* (i.e., the objects RED-UNIT and BLUE-UNIT are subclasses of COMMUNICATING-OBJECT).

A few objects have behaviors that can apply to both Red and Blue forces (currently they exist for Blue only). Most of these are involved in air operations, which have been modeled only at a very general level in TWIRL. They include:

TACC. Tactical air control centers; they initiate air reconnaissance and offensive air support operations, and communicate mission results to the appropriate Red or Blue ground force command element.

AIRBASE. Airbases that launch air missions as requested by TACCs.

MISSION. Aircraft assigned for reconnaissance or attack.

Objects on the Red side include:

RED-UNIT. The broadest generic class, representing all Red units. Only very general, default behaviors and attributes for Red units are stored here. RED-UNIT is a subclass of both COMMUNICATING-OBJECT and MOVING-OBJECT.

DIV. A subclass of RED-UNIT, representing the command elements and other specified units of Red divisions (in general). TWIRL contains three command elements of a single Red division: DIVTAC, DIVMAIN, and DIVREAR.

DIVTAC. The forward tactical division headquarters, which controls the advance to the river and the crossing operations of the two first-echelon (assault) regiments.

DIVMAIN. The fully staffed operational division headquarters; also the two second-echelon maneuver regiments, and organic and attached division, Army, and front fire and combat support units not otherwise identified individually.

DIVREAR. The administrative and logistical support headquarters of the division.

REGT. A subclass of RED-UNIT, representing the headquarters (CP) of the first-echelon (assault) motorized rifle regiment of the division (reinforced), its two second-echelon maneuver battalions, and organic and attached subordinate fire and combat support units not otherwise identified individually.

ASSAULT-UNIT. A subclass of RED-UNIT, assigned to force the river crossing operation; ASSAULT-UNIT includes battalions (ADV) and companies (FWD).

ADV. A subclass of ASSAULT-UNIT, representing first-echelon (assault) motorized rifle battalions of a first-echelon regiment (reinforced); they act as advanced guard for the main body of the regiment.

FWD. A subclass of ASSAULT-UNIT, representing assault companies of a first-echelon battalion acting as forward detachment.

ARTILLERY. A subclass of RED-UNIT, representing artillery units (in general) supporting the advance through artillery fire (both barrage and directed).

DIVARTY. A subclass of ARTILLERY, representing organic division artillery assets, plus those attached from Army and front for the river crossing operation. We postulate that the division will control the equivalent of two artillery regiments.

REGTARTY. A subclass of ARTILLERY, representing all organic and attached artillery assets under operational control of a first-echelon (assault) regiment. We postulate that each assault regiment will control the equivalent of two artillery battalions.

BNARTY. A subclass of ARTILLERY, representing attached artillery assets under operational control of a first-echelon battalion of an assault regiment. We postulate that each assault battalion will control the equivalent of two artillery batteries.

ENG. A subclass of RED-UNIT, representing engineer units assigned to provide route and river crossing site reconnaissance and other preparations, augmented to operate as crossing control teams. We postulate the equivalent of one reinforced engineer reconnaissance platoon for each assault regiment crossing sector.

ELECTRONIC-WARFARE-UNIT. A subclass of RED-UNIT, which includes units with active and passive EW functions.

DFERS. A subclass of ELECTRONIC-WARFARE-UNIT, which performs passive functions and represents COMINT and ELINT units with direction-finding capability.

EH. A subclass of DFERS, which represents high-frequency (HF) radio intercept and direction-finding platoons organic to division reconnaissance battalions. We postulate that a division will possess only one such platoon.

EV. A subclass of DFERS, representing very high-frequency (VHF) radio intercept and direction-finding platoons organic to divisional reconnaissance battalions. We postulate that a division will possess two of these platoons, one to support each regimental crossing sector.

JAMMERS. A subclass of ELECTRONIC-WARFARE-UNIT. JAMMERS are active EW units.

JH. A subclass of JAMMERS, representing HF radio-jamming companies attached to the division for the river crossing operation. We postulate that the division will be allocated one such company.

JV. A subclass of JAMMERS, representing Army VHF radio-jamming companies attached to the division for the river crossing operation. We postulate that the division will be allocated two of these companies, one to support each regimental crossing sector.

Objects on the Blue side include:

BLUE-UNIT. The broadest generic class, representing all Blue units. Only very general, default behaviors and attributes for Blue units are stored here. BLUE-UNIT is a subclass of both COMMUNICATING-OBJECT and MOVING-OBJECT.

BLUE-DIV. A subclass of BLUE-UNIT, representing the parent division (and all superior Blue ground force elements) of the defending Blue regiment. This object enters the simulation only to provide the source of the message to the defending mechanized infantry regiment ordering it to move into defensive positions along the river and, once there, to issue a subsequent tactical alert order.

BLUE-TACC. The Air Force tasking element that schedules (frags) air missions in response to self-generated or ground force requirements.

BLUE-AIRBASE. The aircraft sortie-generation capability available to respond to BLUE-TACC frags.

BLUE-AIRMISSION. The response (i.e., a specific number of aircraft assigned a specific task at a specific time) of BLUE-AIRBASE to a BLUE-TACC frag.

BLUE-FIRING-UNIT. A subclass of BLUE-UNIT, representing the artillery units that attempt to slow Red's movement through directed artillery fire. (Because Blue's artillery units behave differently from Red's, they were not set up as subclasses of a more general artillery class.)

ARTYBN. A subclass of BLUE-FIRING-UNIT, representing the senior artillery CP/fire direction center (FDC) for all the organic and attached artillery (taken to be the equivalent of three batteries) supporting the defending mechanized infantry regiment; it also represents the heavy artillery battery.

ARTYBTRY. A subclass of BLUE-FIRING-UNIT, representing organic or attached medium artillery batteries supporting the defending mechanized infantry regiment. We postulate the regiment controls two such batteries.

MECH-REGT-HQ. A subclass of BLUE-UNIT, representing the command element and CP functions of the defending mechanized infantry regiment. In TWIRL, MECH-REGT-HQ1, an instance of MECH-REGT-HQ, decides the appropriate response (EW or fire support) to targets nominated by direction-finding units.

MECHBN. A subclass of BLUE-UNIT, representing the class of maneuver elements organic to the defending Blue regiment. Four instances of MECHBN exist in TWIRL, but once deployed in their defensive positions, they currently play no further role in the simulation.

DFPLT. A subclass of BLUE-UNIT, representing the class of EW units capable of intercepting and direction-finding all tactical communication nets in use by the attacking Red division. TWIRL contains two instances of DFPLT.

JAMCO. A subclass of BLUE-UNIT, representing the class of active EW units capable of jamming receivers on any of the tactical communication nets in use by the attacking Red division.

Objects used for representing communication channels and networks include:

CHANNEL. The class of communication media over which objects can communicate within the simulation. Its subclasses include BROADCAST-CHANNEL and POINT-TO-POINT-CHANNEL.

BROADCAST-CHANNEL. The class of communication networks over which objects can broadcast messages simultaneously to other objects in the network (which may include the enemy listening in).

POINT-TO-POINT-CHANNEL. The class of direct, point-to-point communication channels.

Objects used for representing "terrain" include:[6]

TERRAIN-OBJECT. The only terrain objects represented in TWIRL are manmade objects, i.e., bridges and road junctions. (Obviously one could introduce such objects as rivers, streams, lakes, hills, roads, obstacles, minefields, flatlands, etc.) Basically, a terrain object "knows" how to determine whether a unit can cross it and either allows the unit to cross or notifies it of its current status.

BRIDGE. A subclass of TERRAIN-OBJECT, representing the class of bridges that exist on Red's route to the major river crossing. Bridges can be destroyed by Blue air interdiction, delaying Red's advance.

ROAD-CROSSING. A subclass of TERRAIN-OBJECT, representing the class of road junctions in the road network that Red's units will be crossing en route to the river. Road junctions can be destroyed by Blue air interdiction, delaying Red's advance.

And finally, two auxiliary objects serve primarily as computational objects:

MATHEMATICIAN. The MATHEMATICIAN does most of TWIRL's computations, e.g., computing distances traversed and updating locations of objects.

PHYSICIST. The PHYSICIST determines the effects of various physical phenomena such as bomb explosions and artillery fire.

Figure 1 presents a ROSS hierarchy of the main objects in the TWIRL simulation. (This is not a military command hierarchy!) Each object represents a subclass of its parent classes (e.g., ASSAULT-UNIT is a subclass of RED-UNIT, which, in turn, is a subclass of both MOVING-OBJECT and COMMUNICATING-OBJECT). Color Plate 1 (in the center section of this report) provides a list of the objects (and other features) that are displayed graphically.

[6] Strictly speaking, all the features included in the TERRAIN-OBJECT class need not be (and are not) actual terrain elements. However, as they affect the simulated events in a similar manner, their inclusion is convenient.

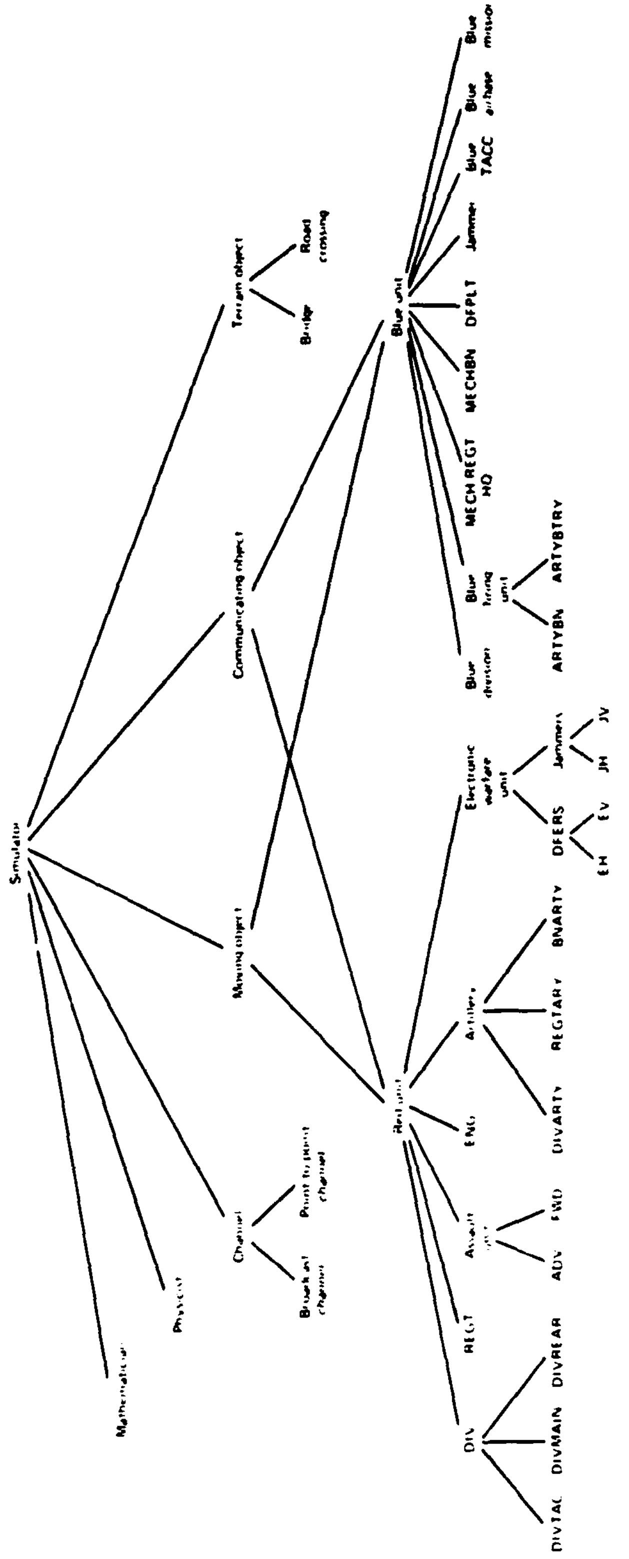


Fig. 1—TWIRL hierarchy of objects

III. REPRESENTING COMMUNICATIONS

Representing the communications between objects is quite straightforward in ROSS. Objects communicate by sending messages to one another. However, real-world communications can be quite complex, exhibiting a variety of characteristics and behaviors. For example, there are different types of communication techniques (e.g., voice, teletype, Morse code), different types of signals (e.g., telephone, radio: AM, FM, SSB), different frequencies, etc. Also, when a message is sent, many things can happen to it en route to the intended receiver, e.g., noise on the channel can garble the message, jamming on a particular frequency can prevent receipt of the message, or other objects can be listening in and also receive the message, unbeknownst to either the sender or the intended receiver.

To accommodate such possibilities and complexities, we created a ROSS object called CHANNEL to represent the class of all communication media. Instances of CHANNEL represent particular communicating channels, each with its own set of characteristics (attributes). When an object sends a message to another object, the message always passes through a communication channel. The channel (like any other ROSS object) has a set of behaviors to respond to various situations (messages sent to jammed or dead objects, enemies listening in, etc.).

The CHANNEL object allowed us to effectively model the communications process. It also suggested a particular strategy for modeling processes within an object-oriented framework, namely,

- To model a set of effects that seems external to the other objects being modeled, create an object to represent those effects and define appropriate behaviors for executing them.

Although we applied this principle in the development of SWIRL, in particular to create objects representing a scheduler (e.g., to determine proximities of objects), a mathematician, and a physicist (e.g., to determine the effects of electromagnetic pulses), its full impact came in the TWIRL development, primarily in the modeling of communications and combat, as we shall see. With a little imagination, one can readily envision creating objects to represent weather, geographical entities, functional entities (segmenting an object into several objects representing the different functions of the object), plans and procedures, hypotheses and beliefs, and even alternative futures.

THE CHANNEL OBJECT

The top-level object in TWIRL is called SIMULATOR. We tell that object to create a new generic (class) object called CHANNEL:

```
(1)  (ask SIMULATOR create generic CHANNEL with
                       normal-delay        0
                       jammed-delay        30).
```

The two attributes specify any delays involved in sending a message, e.g., if the intended receiver is jammed, the message is delayed 30 minutes. (The 30-minute delay, an arbitrarily selected value, represents the time interval necessary for the sender to find another channel, retransmit the message on a new channel, and have the message get through to the intended receiver. We could, of course, model this process explicitly, but for simplicity we represent it here as a delay.)

Next, we define a behavior for CHANNEL that allows an object to send a message through a channel to a receiver:

```
(2)  (ask CHANNEL when receiving
                           (send from >sender to >receiver +message)
            (if (member receiver (~your units-jammed))
             then (~requiring (~your jammed-delay) minutes
                                   tell ~the receiver ~that message)
             else (~requiring (~your normal-delay) minutes
                                   tell ~the receiver ~that message))).
```

When a channel object (any member of the CHANNEL class) receives a message such as "(send from regiment1 to artillery5 begin preparatory fire)," the channel object will first test if artillery5 (the receiver) is a member of the channel's list of units-jammed. If artillery5 is not jammed, the message "begin preparatory fire" will go through without delay; otherwise, there will be a 30-minute delay.

Before creating particular channel instances, we first create a subclass of CHANNEL representing broadcast networks:

```
(3)  (ask CHANNEL create generic BROADCAST-CHANNEL)
```

with the behavior:

```
(4)  (ask BROADCAST-CHANNEL when receiving
                           (broadcast from >sender +message)
            (loop for friend in (~your friendly-listeners) do
               (~you send from ~the sender to ~the friend ~that message))
            (loop for enemy in (~your enemy-listeners) do
               (~you send from ~the sender to ~the enemy ~that message))).
```

Thus, when a broadcast-channel broadcasts a message, it sends it to all friendly listeners on the channel as well as any enemy listeners. (The "send ..." messages used in (4) trigger the behavior defined in (2).)

And, finally, we create a particular channel:

```
(5)  (ask BROADCAST-CHANNEL create instance CHANNEL1 with
            frequency              vhf
            type                   voice
            used-for               command
            side                   red
            friendly-listeners     (regt1 regtarty11 regtarty21 divarty1)
            units-jammed           nil
            enemy-listeners        nil).
```

Thus, CHANNEL1 is an instance of BROADCAST-CHANNEL, which is a subset of CHANNEL. CHANNEL1 has an initial set of attributes as defined in (5). These attributes (as well as everything else in ROSS) can be dynamically modified by the user or as the result of actions occurring in a simulation run. Also, since BROADCAST-CHANNEL is a subset of CHANNEL, CHANNEL1 inherits the two attributes defined in (1) above, as well as the behaviors defined in (2) and (4), so it knows how to broadcast a message and send it to particular receivers.

DISPLAYING COMMUNICATIONS

In the TWIRL development, we were particularly concerned with enabling users to understand the communication and EW activities that go on in a battle. We wanted to display both graphical and textual information about communications and EW. The graphics facility we developed for TWIRL significantly extended the capabilities of our SWIRL graphics, providing techniques to portray communications, jamming, and artillery fire. Much of the work was experimental, trying alternative displays with different colors, highlights, menus, etc., and getting feedback on the various alternatives.

Color graphics represents a critical component of TWIRL. Unfortunately, it must be seen in a live demonstration to appreciate it and understand its impact. Nevertheless, we include several snapshots of a TWIRL simulation to give the reader at least some idea of the graphics facility. Color Plate 2 shows a TWIRL snapshot of Red communications during Red's march toward the river. The sender of the message is displayed in yellow; the arrows indicate the receivers to whom the message is being sent (in this case, it is being sent over a broadcast channel); and those objects that actually receive the message are displayed in orange. The message is printed in the lower left corner. The various times in the top part of the photo specify a "raw" time (in seconds) since the start of the simulation, an "operational" time relative to Red's projected crossing of the river at H-hour, a "day" time indicating time of day, and a "simulation" time relative to the start of the simulation.

At the same time the communications are shown schematically on a color graphics monitor, they are explained in more detail in a textual display on an adjacent terminal. The following examples of textual display are excerpted from a TWIRL simulation:

```
*****  BROADCAST MESSAGE SENT  *****
     Time:       H-6.0
     Channel:    red channel1
     Sender:     red divarty1
     Receivers:  (divarty1 regtarty21 regtarty21 regt1)
     Message:    (divarty1 beginning to move from (71.0 45.0) at time 3600)

***  POINT-TO-POINT MESSAGE SENT  ***
     Time:       H-5.917
     Channel:    red channel5
     Sender:     red regt2
     Receivers:  divtac
     Message:    (regt2 stopping at (67.25 51.0) at time 3898)
```

```
***  POINT-TO-POINT MESSAGE SENT  ***
     Time:        H - 1.8
     Channel:     blue channel6
     Sender:      blue mech-regt-hq1
     Receivers:   artybn1
     Message:     (open fire on bnarty211 at (12.75 38.0) with 1 unit for 15 minutes)

*****  BROADCAST MESSAGE SENT  *****
     Time:        H - 1.75
     Channel:     blue channel2
     Sender:      blue mech-regt-hq1
     Receivers:   (jamco1)
     Message:     (execute jamming on channel1 against adv21 at (12.25 43.0))
```

Since all messages flow through channel objects, these objects are responsible for printing the information. Note that the Time entry is stated as "operational" time in hours relative to H-hour, but the Message entry (in the first two messages) shows time as "simulation" time in seconds (from the start).

IV. REPRESENTING COMBAT

Because we were primarily interested in simulating and displaying EW, the present version of TWIRL makes no attempt to represent maneuver unit combat or the effects of air attack on such ground forces. The primary combat interactions modeled included EC (jamming), artillery fire, and some limited air operations. At the conclusion of this section, we will discuss briefly how maneuver unit combat and close air support might be treated.

ELECTRONIC WARFARE (DIRECTION FINDING, JAMMING)

Jamming refers to the process of electronically blocking objects from receiving messages. Direction finders attempt to intercept enemy communication activities on specific channels and frequencies in order to locate enemy units. In the TWIRL simulation, when the Blue direction finders learn that Red units are stopping (and will be stationary for some time), the Blue commander is notified about the enemy unit. The Blue commander then decides what to do, e.g., to jam or fire artillery.[1] In this section, we look at the process of jamming as it evolved into TWIRL code.

In behavior (4) above (Section III), a broadcast-channel object will send messages to all friendly listeners on the channel, as well as to any enemies that may be listening in. How are direction finders (the enemy listeners) placed on the channel's list of enemy-listeners?

We define an object called DFPLT, a direction-finder platoon, to represent the aggregate functions involved in direction finding. When the Blue commander, called MECH-REGT-HQ1, sends out a "go on alert" message to its units, the Blue units set their "status" attribute to be "alert." The DFPLT does one other thing:

```
(6)  (ask DFPLT when receiving (go on alert)
                    (`requiring 5 minutes set your status to alert)
                    (`you plan after 5 minutes listen)).
```

That is, it also schedules itself, after a 5-minute delay (modeling the delay required to activate its equipment), to start listening:

```
(7)  (ask DFPLT when receiving (listen)
                    (loop for channel in (`your enemy-channels) do
                       (tell `the channel direction finder `me listening))).
```

Thus, the DFPLT sends a message to each of the enemy-channels it knows about (e.g., various frequencies it can listen in on), notifying the channel that it is listening. The receiving channel-object then records the fact that the direction finder is listening:

```
(8)  (ask CHANNEL when receiving (direction finder  -df listening)
                    (`you add `the df to your list of enemy-listeners)
                    (print-df-listening df myself)).
```

[1] Obviously, the commander could decide to do both (jam and strike). At times there are benefits to be gained from this tactic; however, our objective was to demonstrate representative capabilities, not wide latitude in options.

The channel adds the direction finder to its list of enemy-listeners. The call on the function "print-df-listening" results in a textual display, such as

```
*** DIRECTION FINDER LISTENING ***
    Time:       H - 2.02
    Listener:   blue dfplt1
    Channel:    red channel1     Type: broadcast-channel
```

When a direction finder subsequently hears a message, the graphics will also depict this information. An example of this is shown in Color Plate 3.

Let's consider a concrete example. We first create a particular DFPLT:

```
(9)  (ask DFPLT create instance DFPLT1 with
           position          (-1.0 41.0)
           enemy-channels    (channel1)
           track-plan        ((advance to (1.5 39.0) arriving at 4.97 hours))).
```

Suppose DFPLT1, after arriving at its final position, receives a "go on alert" message. Behavior (6) is triggered and, after 5 minutes, DFPLT1's "status" (a new attribute is created for DFPLT1) would be set to "alert" and DFPLT1 would send itself a message to "listen." This would trigger behavior (7), which would result in the message "direction finder DFPLT1 listening" being sent to CHANNEL1. This message, in turn, triggers behavior (8), which causes CHANNEL1 to add DFPLT1 to its list of enemy-listeners. Now, when a Red unit sends a broadcast message through CHANNEL1, behavior (4) is triggered, which results in CHANNEL1 forwarding the message to all friendly listeners as well as DFPLT1, an enemy-listener.

A DFPLT, as it listens in, looks for those messages that indicate that an object has stopped its movement (and will be stationary for a while). When it hears such a message, it notifies the Blue commander MECH-REGT-HQ1:

```
(10)  (ask DFPLT when receiving
                         (>unit stopping at >position +anything)
          (if (and (eq (~your status) 'alert) (is-not-friendly unit))
           then (~requiring 15 minutes broadcast to mech-regt-hq1
                      that potential target ~the unit at ~the position))).
```

(Note that the broadcast message above is slightly different from the one shown in (4). It is directed to a particular member on the broadcast channel.)

If the MECH-REGT-HQ1 decides to jam the unit (MECH-REGT-HQ behaviors are discussed in Section V), he will send a message to his jamming company to execute jamming against the unit on the appropriate channel. This would trigger the jamming company to execute the following behavior:

```
(11)  (ask JAMCO when receiving
                    (execute jamming on >channel against >enemy at >position)
           (~you set your jamming-status to jamming)
           (~you decrement your available-capacity by 1)
           (tell ~the channel ~me jamming ~the enemy)
           (~you add ~the enemy to your list of targets-serviced)).
```

The message sent to the channel would trigger the behavior:

```
(12)  (ask CHANNEL when receiving (>jammer jamming >unit)
              (~you add ~the unit to your list of units-jammed)
              (print-jamming jammer unit myself)
              (tell ~the unit your communications is jammed)).
```

The channel-objects keep track of the units jammed and will delay messages sent to jammed units (see behavior (2)). The call to the function "print-jamming" results in a textual display, such as:

```
*****     JAMMING     *****
        Time:        H - 1.75
        Jammer:      blue jamco1
        Channel:     red channel1     Type: broadcast-channel
        Jammee:      red fwd111
```

At the same time this information is printed on a terminal, the graphics screen will show the particular jammer JAMCO1 jamming unit FWD111. A barrier is placed around FWD111 to indicate that messages would not get through to it on CHANNEL1. The graphics indicating a barrier to communications consist of a bracket on each side of the unit's military symbol. The bracket's line is sawtoothed and color-coded. Color Plate 4 shows a Blue jammer jamming four Red units. Color Plate 5 shows a subsequent Red message being broadcast. Those Red objects that are jammed do not receive the message (i.e., they do not show up as orange).

In summary, the jamming process in TWIRL consists of objects (direction finders) listening in on communications of the opposing side (through channel-objects), commanders being notified, jamming companies being ordered to jam, and the jamming objects having messages delayed.

ARTILLERY FIRE

As an alternative, or addition, to jamming, the Blue commander may decide to command artillery to fire on Red targets. In Section V, we will see how the Blue commander decides whether to fire or jam. Here, we describe how artillery fire is represented in TWIRL.

To initiate artillery fire, the Blue commander sends a message to a BLUE-FIRING-UNIT (an artillery battalion or artillery battery) to open fire:

```
(13)  (ask BLUE-FIRING-UNIT when receiving
                   (open fire on >target at >position with >n units for >m minutes)
              (~you decrement your available-capacity by ~the n)
              (~you add ~the (list target n) to your list of assignments)
              (print-firing myself target)
              (tell physicist ~me assigning ~the n units to ~the target for ~the m minutes)
              (~you plan after ~the m minutes stop firing on ~the target)).
```

The call to the function "print-firing" results in a textual display, such as:

```
XXXXX   BLUE OPENING FIRE   XXXXX
            Time:           H - 1.75
            Firing Unit:    artybtry1
            Target:         bnarty121
```

Also, the color graphics screen will update to show artillery fire directed toward the target from the firing unit. (Color Plate 6 shows an example of two artillery units firing at two targets. Color Plate 7 provides some examples of Red's combat activities, including artillery fire and jamming.) The "stop firing ..." message (the last action in (13)) will also subsequently result in a textual display indicating that the artillery fire has stopped, with the color graphics screen updated accordingly.

The PHYSICIST serves primarily as an intermediary between artillery firing objects and targets (similar to the way CHANNEL objects act as intermediaries between communicating objects). The message to the PHYSICIST in (13) results in the PHYSICIST notifying the target that it is under fire. The PHYSICIST also computes any damage that results from artillery fire. There are, of course, many alternative methods for calculating damage. In TWIRL, damage consists of attrition and delay. Each combat unit in TWIRL is assigned an initial strength (e.g., an artillery unit would have a specified number of guns). Attrition would reduce the strength available according to the PHYSICIST's calculations. Delay would simply modify a unit's track plan.

There are also many options for deciding when to compute damage. For simplicity, the PHYSICIST computes damage once, after the artillery ceases fire on the target:

```
(14)  (ask PHYSICIST when receiving
                    (>arty ceasing fire on >target with >n units after >m minutes)
            (~you tell ~the target to decrement your current-capacity by
              !(~you determine attrition to ~the target resulting from
                          artillery fire of ~the n units for ~the m minutes))
            (~you tell ~the target to plan after
              !(~you determine delay to ~the target resulting from
                          artillery fire of ~the n units for ~the m minutes)
              seconds resume plan)
            (tell ~the target artillery from ~the arty has stopped)).
```

(As an alternative to calculating damage all at one time, the PHYSICIST could calculate damage every minute or even every second, if desired.) Calculations for attrition and delay are based on volume of the artillery fire and duration of firing.[2]

AIR OPERATIONS

A TACC object initiates air operations by commanding (sending messages to) airbases to scramble aircraft. In TWIRL, a TACC can assign reconnaissance missions (either to cover specific locations or to survey an extended area) and ground attack missions (for interdiction attacks on bridges and road junctions). As an example of the former, suppose we wanted Blue's TACC1 (an instance of TACC) to periodically initiate surveillance missions over a

[2] The volume and duration of fire can be measured in any units suitable for the analyst's purpose.

particular area. In TWIRL, this would look like:

```
(15)  (ask TACC1 when receiving
                         (execute periodic surveillance over >destination)
            (~you send point-to-point to !(~your airbase)
                  fly surveillance recon over ~the destination)
            (~you plan after 5400 seconds
                  execute periodic surveillance over ~the destination)).
```

This behavior causes TACC1 to command its airbase to initiate an area surveillance mission every 90 minutes. The last action in (15) resembles a recursive function call. After a 90-minute delay, the same behavior is executed again. The message to the airbase causes a reconnaissance aircraft to be scrambled and directed toward the desired destination.

Blue interdiction operations begin when TACC1 is notified by one of its reconnaissance aircraft that enemy units are moving toward the river (Color Plate 8 shows aircraft sending message to TACC1, which is offscreen to the left), i.e.,

```
(16)  (ask TACC1 when receiving
                         (red units heading toward the river)
            (~you plan after !(~your command-delay) seconds
                  send point-to-point to !(~your airbase)
                  attack bridge1 at position (40.9 42.0)
                  with 2 aircraft arriving in 30 minutes)
            (~you plan after !(~your command-delay) seconds
                  send point-to-point to blue-div1
                  commence advance to block red operation)).
```

TACC1 commands its airbase to initiate an attack on BRIDGE1 (a bridge that several of Red's units will need to cross) and at the same time notifies Blue's ground operations commander BLUE-DIV1 to begin ground operations to block Red's advance.[3]

Currently, aircraft can be assigned to attack bridges and road junctions. An airbase, upon receiving an "attack ..." message, formulates a mission consisting of one or more aircraft and a scheduled plan of operations, e.g., actual time of takeoff, time of arrival over target, etc. The bombing component triggers the behavior:

```
(17)  (ask MISSION when receiving (bomb >target)
            (tell physicist ~the target being bombed by
                  !(~your number-of-aircraft) aircraft)
            (print-bombing myself target)
            (~you plan after !(~your bomb-delay) seconds next event)).
```

Once again, the PHYSICIST serves as an intermediary to calculate the effects of bombings on targets. The call to the function "print-bombing" results in a textual display, such as:

[3] In practice, the TACC would not directly order ground units into action. For simplicity, we have short-circuited the path that air intelligence data would normally follow to ground force command channels, where it could then influence or trigger ground force commanders' actions.

```
****    BOMBING OCCURRED    ****
             Time:        H-5.5
             Mission:     blue airmission2
             Planes:      2
             Target:      bridge1
             Position:    (40.9 42.0)
             Capacity:    0 (totally out)
```

(Color Plate 9 shows the graphics output of this action.)

Suppose TACC1 initiates an attack on BRIDGE1. When the attack occurs, the PHYSICIST is notified, and it computes the resulting damage to the bridge. Each bridge has an associated "capacity" indicating its current operational strength (measured in percentages). If a bridge is at 100 percent capacity, it is fully operational; 0 percent capacity indicates the bridge is totally destroyed; values between 0 and 100 percent indicate fractional damage. No moving object (Red or Blue unit) that approaches the bridge can cross the bridge unless the bridge is sufficiently operational to allow the unit to cross. This calculation should be a function of the type of unit crossing and the characteristics of the bridge itself. Currently in TWIRL, however, the bridge must be at full capacity to allow a unit to cross it, i.e.,

```
(18)  (ask MOVING-OBJECT when receiving (at >bridge)
                (if (lessp (ask ~the bridge recall your capacity) 100)
                 then (~you broadcast to all that stopped at obstacle !bridge
                                 and repairing)
                       (~you set your velocity to (0.0 0.0))
                       (setq repair-time (ask mathematician compute repair
                                 time for ~the bridge))
                       (~you plan after ~the repair-time seconds tell physicist
                                 ~the bridge is repaired)
                       (~you plan after ~the repair-time seconds resume plan)
                else (~you resume plan))).
```

In this behavior, the unit approaching the bridge repairs it. The MATHEMATICIAN calculates the repair time. One could make this process more complicated by allowing only certain types of units to repair bridges. Those units that cannot repair bridges would need to summon the repair units, perhaps causing further delays to planned movements.

Currently, offensive air support operations in TWIRL are initiated only against bridges and road junctions. It is easy to envision how this could be extended to allow attacks on Blue and Red units as well. Basically, one could still use behavior (17), but the PHYSICIST behavior for determining damage would need to consider the type of target hit and would compute damage differently for the different types of objects.

The ability to provide air attacks on enemy units would give the Blue commander yet a third option for dealing with enemy targets (in addition to jamming and artillery fire). The Blue commander could send messages to its TACC to initiate air interdiction, just as it sends messages to its jamming companies to jam and to its artillery units to fire.

Plate 1—TWIRL graphics legend

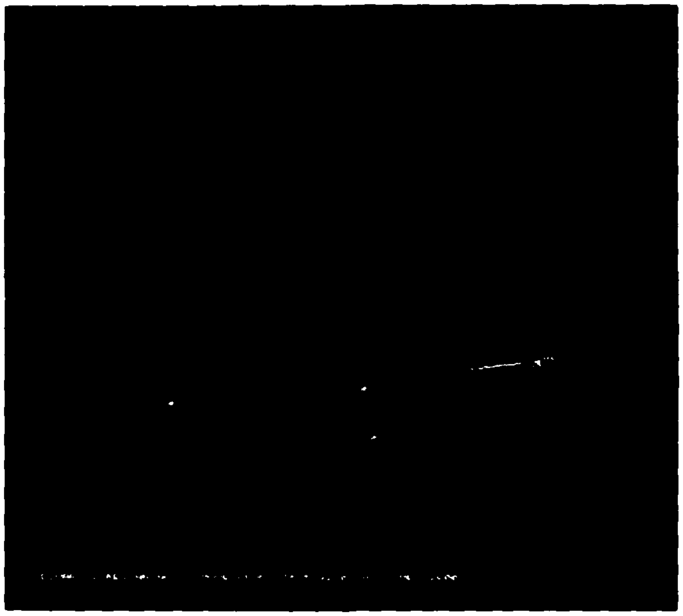


Plate 2—Communications display in TWIRL

Plate 3—Direction finding

Plate 4—Jamming communications

Plate 5—Jammed communications

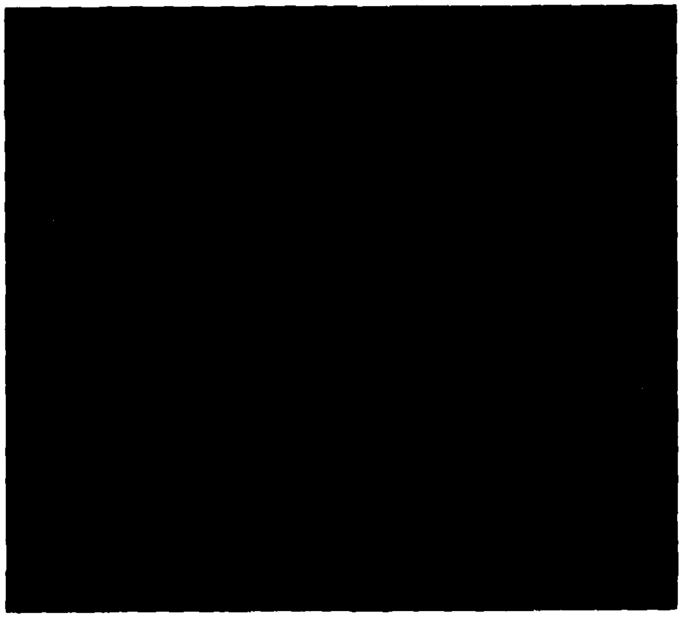


Plate 6—Blue artillery fire

Plate 7—Red artillery fire and jamming

Plate 8—Air surveillance

Plate 9—Air strike

Plate 10—Simulation at H – 7:00

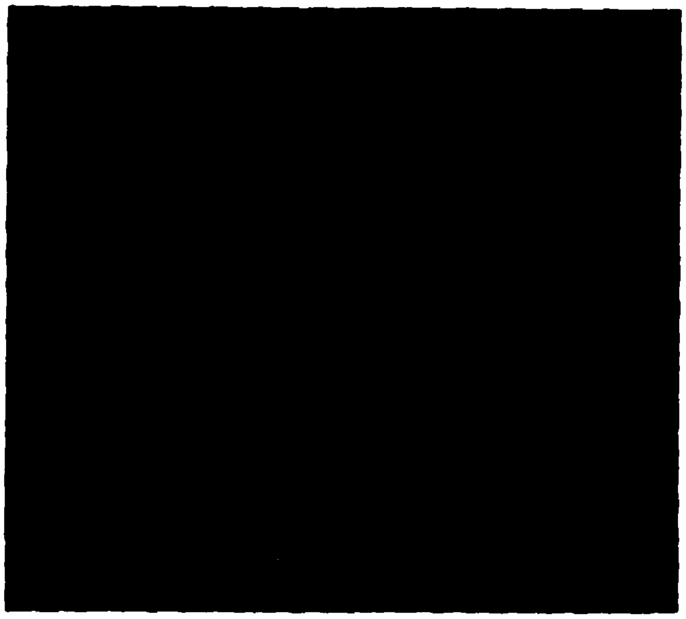


Plate 11—Simulation at H−5:45

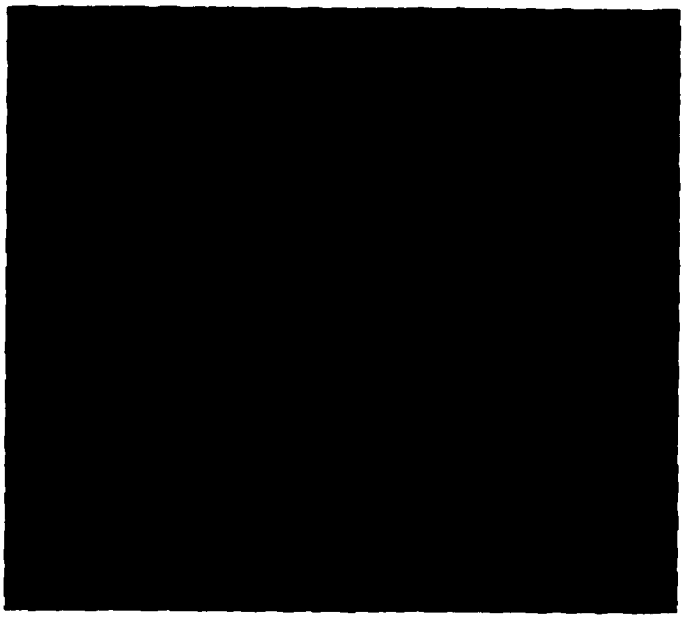


Plate 12—Simulation at H−4:30

Plate 13—Simulation at H – 3:30

Plate 14—Simulation at H – 1:00

Plate 15—Simulation at H−0:00

Plate 16—Simulation at H+0:30

ADDITIONAL FEATURES

Currently, when the track plans of opposing Red and Blue ground forces bring the forces into close proximity with each other, no interaction occurs and each proceeds on its original plan as if the other were not there. This situation could not be tolerated if it was essential to simulate maneuver unit combat activities. A simple example will illustrate how one might approach the simulation of maneuver unit combat in TWIRL.

Each unit might be assigned attributes that specify its initial strength, the range at which it could detect and engage various types of enemy units, the attrition it might inflict and suffer in a given engagement, how long the engagement might last, and how the unit's subsequent behavior might be modified. Once a ground unit approaches to within detection range of an enemy unit, the PHYSICIST could be notified to calculate the mutual attrition rates. The resulting rates and directions of movement for the two combatants could be determined (as a function of force ratio, for example) periodically to provide a record trace of the FLOT (forward line of own troops). Behaviors specifying the circumstances under which a unit would break contact, hold, fight a withdrawal, call for artillery or other fire support, etc., could also be included to expand the repertoire of simulated activities.

Inclusion of on-call artillery or aerial fire support requires a mechanism to originate the request. As alluded to above, an engaged ground unit could originate the request (i.e., send a message through the proper channels) when, for example, its strength, the force opposing it, or its retrograde move rate reached a specified threshold value. The channel could be simulated to whatever level of complexity the analytical problem dictates, including some or all of the ground force and air echelons that are involved in present Service doctrine. Higher ground force echelons could introduce delays for assessment and consolidation of multiple requests, substitute artillery for a requested air strike, or exercise veto power, for example. But once an approved request reaches the Air Force's TACC, the procedure could be similar to that described above: The TACC would decide the number and type of aircraft to commit, the air unit to execute the mission, the ordnance, and the desired time on target, and would then send a message to the airbase to execute the mission.

These simple examples of additional combat interactions illustrate how one could begin to build into TWIRL far more complexity than is incorporated in our prototype simulation.

V. DECISIONMAKING: THE BLUE COMMANDER

AUTOMATING THE DECISIONMAKING PROCESS

As an example of how to model decisionmaking in TWIRL, we consider the Blue commander called MECH-REGT-HQ1 (an instance of the general class of commanders MECH-REGT-HQ (mechanized regiment headquarters)). The Blue direction finder alerts its MECH-REGT-HQ when it determines that a Red unit has stopped at a particular location (see behavior (10) above). This triggers the following behavior:[1]

```
(19)  (ask MECH-REGT-HQ when receiving
                     (potential target >unit at >position)
         (if (~you determine there is an artillery priority for ~the unit)
          then (~you assign artillery to ~the unit at ~the position
                     based on targets to be serviced)
          else (~you assign jammer to ~the unit at ~the position))).
```

The first thing the MECH-REGT-HQ does is determine the artillery priority of the unit. Basically, the MECH-REGT-HQ examines its "target-priority-list" attribute to see if the enemy unit falls into any of the classes of targets listed there. If it does not, the MECH-REGT-HQ decides to assign a jammer to the enemy unit. (This evolves into the MECH-REGT-HQ sending an "execute jamming ..." message to a jamming company triggering the behavior shown in (11) above.)

Next, MECH-REGT-HQ checks to see if there are any enemy targets waiting to be serviced (which might be the case, for example, if there were more targets than available artillery):

```
(20)  (ask MECH-REGT-HQ when receiving
               (assign artillery to >unit at >position based on targets to be serviced)
         (if (~your targets-waiting-to-be-serviced)
          then (assign artillery to ~the unit at ~the position based on
                    target priorities)
          else (~you assign artillery to ~the unit at ~the position))).
```

If there are no other enemy targets waiting to be serviced, the MECH-REGT-HQ assigns artillery to the new target (see behavior (23) below). Otherwise, the artillery priority of the new enemy target is compared with the priorities of the targets waiting to be serviced:

[1] We must reemphasize that the particular behaviors we have developed for TWIRL are exemplary only and not meant to portray realistically the behaviors of any corresponding real-world entity. The goal is to show how one could go about doing so, if desired.

```
(21)  (ask MECH-REGT-HQ when receiving
                    (assign artillery to >unit at >position based on target priorities)
          (if (greaterp (~you determine artillery priority of ~the unit)
                        (~you determine highest-priority of waiting targets))
           then (~you assign artillery to ~the unit at ~the position)
           else (~you add-by-priority ~the unit to your list of
                        targets-waiting-to-be-serviced)
                 (~you assign jammer to ~the unit at ~the position))).
```

If the priority of the new enemy target is greater, the MECH-REGT-HQ assigns artillery to the new target. If it is not, the MECH-REGT-HQ adds the new target to its "targets-waiting-to-be-serviced" list and in the meantime assigns a jammer to the new target. Note that the MECH-REGT-HQ still wants to bring artillery fire to bear on the new target. (The above three behaviors—(19), (20), and (21)—could be combined into one large behavior. We chose to represent them as three to indicate the three decision points.)

Two loose ends remain. First, when, if ever, do the enemy units waiting to be serviced actually get serviced? This occurs when an artillery unit reports that it has ceased firing on its assigned target (artillery fire occurs for particular intervals of time determined by the MECH-REGT-HQ):

```
(22)  (ask MECH-REGT-HQ when receiving
                                         (>artillery ceasing fire on >target)
          (if (~your targets-waiting-to-be-serviced)
           then (setq newtarget (~you determine highest-priority of waiting targets))
                (~you remove ~the newtarget from your list of targets-waiting-to-be-serviced)
                (~you assign artillery to ~the newtarget at
                       !(~you determine position of ~the newtarget)))).
```

Second, the MECH-REGT-HQ has to assign artillery to targets:

```
(23)  (ask MECH-REGT-HQ when receiving
                          (assign artillery to >target at >position)
          (if (~you determine that no artillery are available)
           then (~you consider reassigning artillery with lowest
                      priority target to ~the target at ~the position)
           else (~you send point-to-point to !(~you find best available artillery)
                      open fire on ~the target at ~the position with
                      !(~you determine number of firing units for ~the target) units
                      for !(~you determine duration to fire at ~the target) minutes))).
```

If no artillery is available, the MECH-REGT-HQ considers reassigning a currently firing artillery unit to the new target (see (24) below). Otherwise, it decides to fire on the target and looks for the best available artillery unit (based on available capacity of the artillery and distance from the target). It then sends a point-to-point message to the artillery unit to open fire (triggering behavior (13)). The number of firing elements to assign and the duration of firing are determined by the MECH-REGT-HQ, based on the target type (a list relates target types to number of firing elements and firing duration).

Finally, we show how MECH-REGT-HQ can reassign artillery:

```
(24)  (ask MECH-REGT-HQ when receiving
                          (consider reassigning artillery with lowest priority
                              target to >new-target at >position)
          (setq low-artillery (~you find artillery with lowest target))
          (setq low-target (~you find target with lowest priority))
          (if (greaterp (~you determine artillery priority of ~the new-target)
                        (~you determine artillery priority of ~the low-target))
           then (~you send point-to-point to ~the low-artillery
                              cease fire on ~the low-target)
              (~you assign jammer to ~the low-target at
                          !(~you determine position of ~the low-target))
              (~you send point-to-point to ~the low-artillery
                          open fire on ~the new-target at ~the position with
                          !(~you determine number of firing units for ~the
                          new-target) units for
                          !(~you determine duration to fire at ~the new-target)
                          minutes)
              (~you add-by-priority ~the low-target to your list of
                          targets-waiting-to-be-serviced)
           else (~you assign jammer to ~the new-target at ~the position)
              (~you add-by-priority ~the new-target to your list of
                          targets-waiting-to-be-serviced))).
```

We can summarize the decisionmaking process of the Blue commander as follows: It can assign jammers and artillery to enemy units. It prefers to assign artillery for those enemy units that are on its artillery priority list. It jams those enemy units not on the artillery list. If it prefers to assign artillery to an enemy unit but cannot, it jams the unit but still intends to assign artillery in the future, if at all possible. It will assign artillery to enemy units, based on its priority list, until all of its artillery is in use. Artillery units can be reassigned at any time if targets with higher priorities are detected. When artillery units are freed up, they are reassigned to any enemy units waiting to be serviced.

This behavior is obviously only one of many possible sets of behaviors. For example, additional complexity could be added to the list of possible behaviors by expanding or limiting the number of available artillery tubes per battery, the number and types of artillery rounds, the accuracy of target acquisition and of placing rounds on targets, etc.

Alternatively, instead of automating the Blue commander, a human player may be allowed to make those decisions as they arise within an actual simulation run. This capability is discussed below.

INCORPORATING HUMAN DECISIONS (HUMAN IN THE LOOP)

The object-oriented organization of ROSS provides an excellent environment in which to allow a human to play one of the simulation objects (or, more generally, to allow any number of humans to play any number of simulation objects). The human becomes a ROSS object just as the automated simulation players are ROSS objects, i.e., the human receives and sends messages just like any other ROSS object.

To illustrate this capability, we have developed a prototype facility in TWIRL in which a human can play the role of a Blue commander (the MECH-REGT-HQ1). Recall that a direction finder, when it locates an enemy unit (behavior (10)), notifies the MECH-REGT-HQ1, who in turn decides what to do about the enemy (behavior (19)). We can redefine behavior (19) for our Blue commander MECH-REGT-HQ1 to be the following:

```
(25)  (ask MECH-REGT-HQ1 when receiving
                                (potential target >unit at >position)
           (print-target-menu unit position)).
```

Note that behavior (19) applies to the class of all MECH-REGT-HQ, whereas (25) applies to only one particular instance of that class, MECH-REGT-HQ1. All other Blue commanders still have behavior (19).

When MECH-REGT-HQ1, which a human is now playing, is notified of a potential enemy target, the function "print-target-menu" is called to list the available options. Since the human playing MECH-REGT-HQ1 would probably not know what his options are or what messages he can send, we decided to facilitate the interaction by means of a menu. Figure 2 shows a typical interaction.

The options available to the human in Fig. 2 correspond to particular messages to be sent, or actions to be taken. For example, Option 1, the one actually selected by the human, causes an "open fire ..." message to be sent to ARTYBTRY1, which triggers a behavior similar to (13). Option 2, if selected, would trigger the chosen jamming company to execute behavior (11). Option 3 would result in a "cease fire ..." message being sent, etc. Option 7 would eliminate behavior (25) and result in the MECH-REGT-HQ1 again exhibiting behavior (19).

We should clarify that what we have done in behavior (25) and Fig. 2 is allow a human to decide what to do about potential enemy targets as they occur in a simulation. In a sense, the human is acting as a part of MECH-REGT-HQ1, but not necessarily all of MECH-REGT-HQ1. We have redefined only one behavior (although this behavior (19) can be complex, e.g., possibly triggering behaviors (20), (21), (23) and (24)). Other behaviors for MECH-REGT-HQ1 have not been changed. Thus, the human need not replace an object completely, but may replace only part of the object, for example those behaviors of importance or focus. The human can be allowed to replace as much of an object as desired by redefining the appropriate behaviors.

Allowing a human to play an object, or certain functions of an object, has great utility for training and analysis. For training, this facility allows the trainee to make decisions in a combat simulation and see their impact as the simulation progresses. For analysis, it is possible to make decisions to direct the simulation toward various situations and alternatives and to explore various strategies and tactics, again seeing their impact in the simulation.

```
TIME = H-1.5


 UNIT      POSITION    TARGET    ASSIGNED    TOTAL     AVAILABLE  T_BEG   T_END
                                            CAPACITY
artybn1    (1.0 48.0)  bnarty121     1         2           1       18910   19810
artybtry2  (5.5 51.5)  bnarty211     1         1           0       18960   19860
artybtry1  (5.5 38.5)     —          0         1           1         —       —
jamco1     (1.5 43.5)  adv12        yes        8           6       18300   20100
                       adv11        yes                            19400   21200

Target sited by dfplt1
Target: fwd111      Position: (12.25 38.5)

You may:
            1. assign artillery
            2. assign jammer
            3. cease fire on a target
            4. cease jamming a target
            5. seek information
            6. evaluate Lisp form
            7. have automated blue regiment take over decisionmaking
            8. ignore the object

            Please type the appropriate number:  1

Specify artillery you wish to assign:  artybtry1

  XXXXX   BLUE OPENING FIRE   XXXXX
        Time:           H-1.5
        Firing Unit:    artybtry1
        Target:         fwd111
```

Fig. 2—Human decisionmaking in TWIRL (human responses in boldface)

VI. RESULTS OF AN EXEMPLARY SIMULATION

In Section II, we briefly described the problem domain simulated by TWIRL. Here we will assess the sequence of events produced by the current TWIRL code and indicate where (in that sequence) and how Red and Blue combat actions cause deviations in each others' planned activities. Recall that in TWIRL each object's intended movements and planned combat actions are specified in a *track plan*. In essence, track plans embody each side's operational plans or goals. This analysis of an exemplary TWIRL simulation focuses mainly on how Blue actions cause Red to alter its track plans, which, in turn, cause delays in Red's commencing the assault phase of the operation.

To describe and explain the actions and reactions of the simulated Red and Blue combat units, we refer to specific instances of the classes (generic objects) defined in Section II. Figure 1 illustrated the hierarchical relationships of those classes in the ROSS code. Specific instances of a class (e.g., a certain motorized rifle regiment is an instance of the class of all Red motorized rifle regiments) are given unique identification numbers. These, together with the class name, indicate position in the military command hierarchy. Figure 3 shows the unit numbering system adopted for the Red command structure in the current TWIRL simulation. Note that REGT2 is not simulated in as much detail as REGT1, i.e., no fire support units below the motorized rifle division level (REGTARTY or BNARTY units) are included for REGT2. Their absence introduces no difficulties, as REGT2 plays only the role of a baseline control case. We purposely designed the ROSS code not to allow Blue's disruptive efforts to affect REGT2's scheduled track plans.[1]

Basically, the Red operational plan (collectively defined by the individual Red unit track plans) calls for the major (maneuver and fire support) units to depart from a rear assembly area and move over designated lines of communication (LOCs) toward assault positions near the river. The actual assault crossing is scheduled for a specified H-hour, when all essential units are to be in place and ready. During the approach march, each major assault unit (i.e., REGT1 and REGT2) moves independently, periodically detaching subordinate or attached subunits (e.g., ENG1, DIVEV2), each of which then continues separately toward its own individual assault location. Also, during the approach march, the track plans of the fire support units instruct them to halt from time to time to set up a firing position from which to provide on-call fire support to cover the march routes of the maneuver units. The timing of the halts produces a leapfrogging pattern to ensure continuous fire support coverage during the march. (See Appendix A.)

Once the major maneuver and fire support units reach their forward assembly areas, the subordinate assault and fire support units (e.g., FWD111, BNARTY121) detach and deploy to their assault positions to await the assault order.

In the current TWIRL code, delay in executing a unit's scheduled track plan provides a measure of the collective effectiveness of three means of disruption: air attacks on the LOCs, artillery fire on the units themselves, and jamming of their communications. The consequences to Red if Blue were to employ all three are described below. The Red responsive behaviors are those described in Section IV.

[1] As mentioned earlier, the use of REGT2 as a control facilitates identification of the impact of disruptive measures on the behavior of REGT1. This is particularly useful when viewing the graphic display.

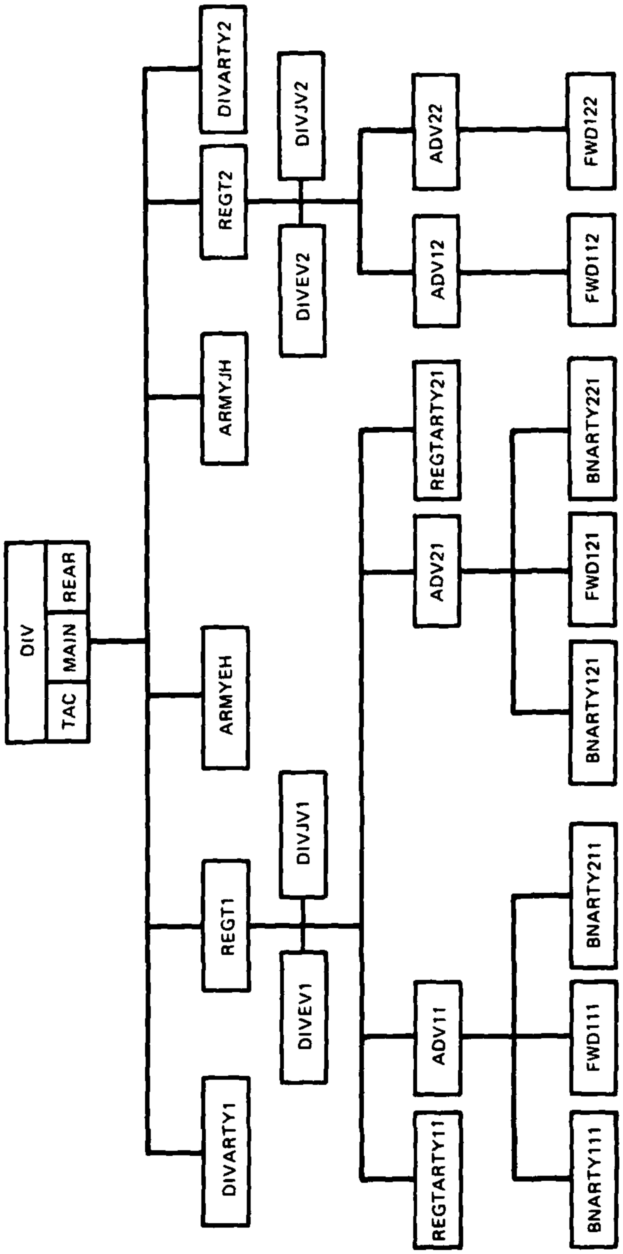


Fig. 3—Simulated Red military command hierarchy

Blue reconnaissance flights discover Red units moving out of the division assembly area (see Color Plate 8). Upon receipt of this information, the Blue air component commander (through his TACC) plans to interdict the likely Red LOCs, choosing two bridges and five road junctions as targets (see, for example, Color Plate 9). Attacks on four road junctions and both bridges create obstacles that, until repaired, are sufficient to stop movement at those points. These obstacles are on routes to be used by DIVTAC, REGT1, DIVARTY1, and REGTARTY21, with the latter two sharing a portion of a single route.

Upon encountering an obstacle, Red unit behavior calls for the unit to stop, clear the obstacle, or repair damaged facilities, using its organic engineer assets. This action is assumed to occur at the rate of restoring 4 percent of capacity per minute (i.e., for damage preventing any movement across the obstacle, full movement is restored in 25 minutes). If several units share the same route, and following units arrive at the obstacle before an earlier unit has cleared it, the new unit's repair capability is added, thereby increasing the repair rate proportionately.

In our example, DIVARTY1 and REGTARTY21 start out on separate routes that merge at one of the damaged road junctions, continue together over a damaged bridge, then diverge on separate routes. DIVARTY1 arrives at the damaged road junction first, finds it damaged, repairs it, and moves on, having suffered a 25-minute delay in its track plan. REGTARTY21, arriving somewhat later, finds the road junction at full capacity and suffers no delay, but the interval between it and DIVARTY1 has now been decreased by 25 minutes.

Upon arriving at the damaged bridge, DIVARTY1 begins repairs and, some 20 minutes later, is joined by REGTARTY21. Their combined repair capacity completes the job in a total of 22 minutes, rather than the 25 minutes it would have taken DIVARTY1 working alone. As movement begins again, a following unit's behavior requires it to remain 5 minutes behind any leading unit. Hence, the bridge repairs cost DIVARTY1 a delay of 22 minutes and REGTARTY21 a delay of 7 minutes. These two units then continue with their track plans.

As a result of the above delays, DIVARTY1 is 47 minutes behind time (H−3:43 vs. H−4:30) in arriving at a temporary halting site from which it was intended to provide on-call fire support for all REGT1 units during their approach march. However, unimpeded units would probably have outdistanced the effective reach of DIVARTY1's guns, as the supported units have widened the interval by some 20 to 25 km. Hence, this mission would probably be unfulfilled unless suitable adaptive behavior were added to Red units to maintain the intended timing among all the advancing Red units.[2] These behaviors were not essential for our purposes, so DIVARTY1 remains in its temporary firing site the scheduled 60 minutes, departing at H−2:43 instead of H−3:30. DIVARTY1 encounters no further obstacles on its approach march, closing its assault firing position at H−0:43, 47 minutes late. The consequences of this late arrival are discussed below.

REGTARTY21 has no en route supporting fire responsibilities and was scheduled to continue directly to its forward assembly area. However, it encounters a bombed road junction further along the way, repairs it in 25 minutes, and closes its assembly area 32 minutes behind schedule, at H−2:43 instead of H−3:15. The original schedule called for REGTARTY21, having arrived at its forward assembly area, to organize two artillery task groups (BNARTY121 and BNARTY221) and to prepare them to move forward in support of the motorized rifle assault units. The first of these (BNARTY121) was scheduled to depart from the forward assembly area at H−2:30, 45 minutes after REGTARTY21 arrived. The current TWIRL code allows this schedule to be met. (However, if that 45-minute period were the minimum time

[2] For this simple example, no attempt has been made to incorporate behaviors in Red units that could sense the developing maltiming among units and attempt to adjust movement schedules to compensate.

required for REGTARTY21 to organize an artillery task group, than BNARTY121's schedule would have slipped here also.) REGTARTY21's offspring (BNARTY121 and BNARTY221) arrive at their respective assault positions on their original schedules. However, the behavior code for REGTARTY21 requires it to spend its scheduled 60-minute residence time in its forward assembly area. Hence, REGTARTY21 does not arrive at its assault position until H–1:13, 32 minutes late.

En route to its forward assembly area, REGT1 encounters two damaged road junctions, repairs each one, and arrives at its assembly area at H–3:10, 50 minutes late. Upon arrival, REGT1 planned to designate and dispatch two engineer units (ENG1 and ENG2) to survey battalion crossing sites and establish crossing control points—ENG1 was to continue forward immediately, while ENG2 was to depart from the assembly area 15 minutes after arrival. Since REGT1 arrives 50 minutes late, ENG1 and ENG2 each depart 50 minutes later than intended. Hence, ENG1 would not complete its mission until H–0:40 instead of H–1:30, and ENG2 would complete its mission at H–0:25 instead of H–1:15.[3] Since REGT1 was to remain at this position monitoring and controlling its units' approach until H-hour, the 50-minute delay in arrival would not prevent it from moving then, as required.

In addition, during REGT1's halt in its forward assembly area, it organizes and deploys its assault companies (FWD111 and FWD121) and battalions (ADV11 and ADV21). With the current TWIRL behaviors, these actions proceed according to the original track plans, with no attempt to shorten the stay in the assembly area to make up lost time. As a result, FWD111, FWD121, ADV11, and ADV21 close their final assault positions at H–1:10 instead of H–2:00 (since REGT1 arrived in the assembly area 50 minutes late).

As DIVTAC moves forward toward the location planned for its interim approach march control command post (CP), it encounters a damaged bridge, repairs it, and arrives at its CP site at H–4:05, 25 minutes late. Its direction-finder units (ARMYEH, DIVEV1, and DIVEV2) were scheduled to depart for their assault positions 27 minutes after DIVTAC's arrival, and the jammer units (ARMYJH, DIVJV1, and DIVJV2) an additional 15 minutes later. All EW units adhere to their original schedules, with no adjustments to make up lost time. Thus, the jammer units supporting REGT1 (ARMYJH and DIVJV1) reach their assault positions 25 minutes later than planned (because of DIVTAC's 25-minute delay)—ARMYJH at H–1:28 instead of H–1:53, and DIVJV1 at H–0:38 instead of H–1:03.

As Red units continue to move into their assault positions, Blue DFPLT1 intercepts their messages, causing MECH-REGT-HQ1 to allocate artillery and jamming against selected targets. An artillery mission against a unit results in the unit being out of action during the mission plus a subsequent recovery period, which depends on the strength and duration of the fire mission. Jamming delays the receipt of, and response to, orders during the duration of jamming.[4] These two mechanisms can disrupt Red's preparations for assault enough to cause REGT1 to delay the assault order. REGT1 plans to issue the assault order at H–0:45, but its behavior requires all designated assault (motorized rifle, artillery, and jammer) units to have reported being in position and ready to assault. The delays described above cause two assault units, DIVARTY1 and DIVJV1, to not be in their assault positions at H–0:45. Therefore, REGT1 does not issue the assault order. REGT1's behavior provides for reassessment of

[3] As the original schedule called for REGT1 to issue the assault order at H–0:45, these delays could have been critical. Behaviors to make the assault order dependent upon timely completion of the missions of ENG1 and ENG2 were not (but could easily have been) included in the TWIRL code.

[4] However, movements scheduled in the unit's track plan will be executed on time, even though incoming communications are jammed. We judged this behavior to be more reasonable than to have such movements held in abeyance in case the presence of jamming *might* be blocking an incoming message that *might* contain revised movement orders.

readiness every 15 minutes thereafter. DIVARTY1 reports ready to assault at H−0:43 and DIVJV1 at H−0:38. Hence, REGT1 issues the assault order at H−0:30, 15 minutes late.

However, one other behavior affects the content of the assault order. If all assault units have unjammed communications, the series of actions commanded by the assault order commences immediately. If jamming currently affects communications to at least one assault unit, the execution time in the assault order is delayed 40 minutes. As explained in Section IV, this delay provides an allowance to account for working around the jamming or adapting to other nonelectronic means of communication. In this scenario, at H−0:30, REGT1 finds that communications with four of its assault units (FWD111, FWD121, ADV11, and ADV21) are jammed, and thus incorporates a 40-minute delay in the execution times contained in the assault order.

In the original schedule, the assault order (issued at H−0:45) would result in ARMYJH and DIVJV1 commencing jamming immediately; DIVARTY1, REGTARTY11, and REGTARTY21 opening barrage fire on Blue positions at H−0:30; and FWD111, FWD121, ADV11, and ADV21 advancing toward the river with BNARTY111, BNARTY211, BNARTY121, and BNARTY221 in support at H−0:15. However, with the delays described above (and given no further Blue disruptive actions), each of these activities would occur 55 minutes later than originally scheduled (15 minutes delay in assault order plus 40 minutes allowance for the presence of jamming).

Thus, with the 55-minute delay, ARMYJH and DIVJV1 should begin jamming at H+0:10. ARMYJH responds, but DIVJV1 does not, as it was taken under fire by Blue ARTYBTRY1 from H−0:23 to H−0:08 and is still out of action at H+0:10.

Both DIVARTY1 and REGTARTY11 open barrage fire on the new assault schedule at H+0:25. But REGTARTY21 was taken under artillery fire (from H−0:58 to H−0:43) of sufficient strength to reduce its firing capacity to zero, and thus it cannot respond to the assault order.

All four motorized rifle assault units move out on the assault as ordered at H+0:40, accompanied by their supporting artillery. The latter set up positions (on the new assault schedule) at H+0:45 and open direct fire on the crossing sites. Two of the supporting artillery units, BNARTY121 and BNARTY211, were under Blue artillery fire from H−1:45 to H−1:30, and the other two, BNARTY111 and BNARTY221, from H−1:00 to H−0:45, resulting in reduced capacity to provide direct fire support for the initial crossings.

In sum, the exemplary TWIRL behaviors for Blue air interdiction attacks produced delays and mistimings among some Red units. Delays to two units proved meaningful in that their arrivals at their assault positions were late enough to require postponing the final assault order. Blue jamming of some assault units' communications caused Red to allow additional time for orders to get to intended recipients, adding to the cumulative delay in executing the assault. Blue artillery fire reduced Red jamming capability beginning shortly before the final assault, thus mitigating a potential Red threat to its own targeting capabilities. In addition, Blue artillery was able to reduce the strength of Red artillery available for preparatory fires and for direct support of the assault unit crossings.

VII. CONCLUDING REMARKS

SIMULATION DEVELOPMENT

A primary objective of our research was to test the utility of the ROSS environment for building combat simulations. In particular, we wanted to design and implement a prototype battle simulation that could be used for experimenting with alternative strategies and tactics for using EW in a combat environment. Through the presentation of approximately 25 examples of TWIRL code (written in ROSS), we have tried to provide an understanding of how one builds a simulation in ROSS, and to show that, in fact, ROSS provides an excellent environment for structuring and understanding combat simulations. Other research projects using ROSS have reached similar conclusions (Dockery, 1982; Gunsch and Hebert, 1983; Nugent, 1983; Conker et al., 1983; Steeb et al., 1984).

The object-oriented message-passing framework of ROSS is well matched to the requirements of a combat simulation such as TWIRL. The discrete components of the simulation map directly onto objects, while communications among the objects map onto messages. The object-oriented organization also conveniently allows one to create objects to model special effects within a simulation (e.g., the CHANNEL and PHYSICIST objects in TWIRL).

Other features of ROSS that enhanced TWIRL's development were ROSS's fairly easy syntax; its readable, English-like appearance, which can be tailored to a user's taste; its interactive capability; its modularity (which made the human-in-the-loop feature easy to implement); and the ease with which one can extend a simulation. This last feature deserves to be particularly highlighted.

Building TWIRL was an incremental process. A few objects were created, some behaviors were designed and tested, and then the simulation was run. (In about three weeks, we had a dozen objects and two dozen behaviors for TWIRL.) Feedback from simulation runs was used to debug the existing behaviors and decide where further detail was required. This led to more objects and more behaviors, and again, further simulation testing. Thus, seeing results of a small, initial simulation fed the knowledge-building process. Gaps and hidden assumptions were discovered and filled in, incorrect behaviors were modified or replaced, etc. This process is in sharp contrast to the way most simulations are developed, where the design process is separate from the implementation. In ROSS, design and implementation occur together and influence each other continually as the system evolves.

The incremental model development process in ROSS has some very useful features. First, an initial, fairly simple, running prototype can usually be developed in a few weeks. From this point onward, a demonstrable system always exists. Second, knowledge is debugged incrementally, in small chunks. Third, having a visual display of simulations, at even general levels, suggests areas where further knowledge should be added (where complexity should be increased and gaps filled in). Finally, having a workable, continually expanding system with visual feedback keeps motivation and interest high.

We must again emphasize what we consider to be a crucial component for work in simulation—color graphics. The ability to see simulations as they are running not only helped us to debug behaviors and verify their interactions, but also enhanced our understanding of the simulation tremendously. Global interactions and patterns are not readily discernible from printed outputs of events and aggregated statistics. The ability to view simulations dynamically significantly enhances the building of them and, we feel, will determine their ultimate utility and impact. All of the underlying schemes for representing knowledge, for executing simulations, and for verification become secondary unless the computer simulation can communicate effectively with the ultimate user. In our work, color graphics displays were a key ingredient in TWIRL's success.

We must present two potential problems or warnings about using ROSS: First, there is the issue of speed and the ability to scale up to larger simulations. Our work with TWIRL did not directly address this problem. ROSS is not computationally fast when run simultaneously with graphic displays. For our TWIRL simulation, however, speed was not a major issue. The compiled TWIRL system containing approximately 140 objects and 200 behaviors ran at acceptable speeds on a standalone, dedicated VAX 11/750. (Appendix B describes the TWIRL computing environment.) When scaling up to larger simulations, however, we will need to address the speed issue. Jefferson and Sowizral (1982), in their work on Time Warps, directly address this problem, proposing a new method for distributing simulations among several computers operating in parallel.

The second issue involves the completeness of ROSS as a programming language. ROSS was developed within the Lisp language environment. When operating in ROSS, one is actually within Lisp (with ROSS loaded). Therefore, when using ROSS, one must be familiar with Lisp. In fact, various Lisp functions must often be used, particularly for arithmetic operations and list processing. Potential users should be aware that learning ROSS also involves learning Lisp.

ANALYTIC CONSIDERATIONS

Another goal of our research was to demonstrate that a TWIRL-like simulation could provide insights into the nature and usefulness of electronic combat. Therefore, we purposely limited the scope of the phenomena explicitly modeled in our river crossing scenario. We chose to focus on the disruptive effects of only a few of the means available to a commander to counter a prospective enemy operation. Three means of attack were included in the simulation: air attacks on lines of communications, artillery fire directed at enemy units, and communications jamming.

All three means of disruption impose a delay on an affected unit in carrying out its intended mission. Hence, in our prototype simulation, the magnitude of delay provides a first-order indicator of the effectiveness of disruption. Typically, delays imposed on affected units force adjustments in the time schedules of their subsequent actions or of coordinated or supporting actions of other units. The total impact of the individual deviations from scheduled track plans comes together where joint, coordinated action occurs. In the river crossing scenario, this coordinated action is the divisionwide direct assault on the river, ordered only after all critical participants have reached their assigned jump-off areas. TWIRL's graphic depiction of disruptive actions and effects, together with the transparency of the ROSS code, allows the individual and cumulative delays to be traced to their sources more easily than do other simulation techniques.

In dealing with delay, the TWIRL prototype does not explicitly model the processes by which delay is generated. That is, delays due to LOC attacks and communications jamming are assigned arbitrary values. To do otherwise would have meant greatly expanding the scope of our simulation in terms of additional objects and behaviors—an investment deemed unnecessary for a prototype demonstration. However, analysis of substantive questions of fire support or electronic combat would require such expansion.

For instance, investigation of differing strategies and tactics of electronic warfare would require adding the behaviors of appropriate actors sufficient to define the functions and instrumentalities of the portion of the enemy command and control apparatus that is of interest. The degree and specificity of detail included in the simulation of the enemy C^3 system would have to match that of the friendly countermeasure strategies, tactics, and systems subject to analysis.

In summary, we believe that the TWIRL prototype has demonstrated the ability to produce and display (textually and graphically) the effects of selected specific combat activities. Moreover, TWIRL provides a convenient mechanism to trace cause and effect as they propagate and multiply through a complex set of interconnected activities. Although the TWIRL prototype was never intended to produce substantive analytical results or insights, we believe that its operation has clearly illustrated that a useful analytical tool can be built upon it by simply adding those behaviors and actors needed to define the operational environment at a level of detail commensurate with the analytic questions of interest.

APPLICATIONS AND UTILITY

We believe that the development of the ROSS language and its application through SWIRL and TWIRL can lead to significant improvements in the analysis of problems characterized by complex interactions among many components. The examples noted below illustrate the range of applicability of the technique and some idea of its utility.

Vulnerability assessment. All systems, particularly large and complex ones such as field forces, are bound to have inherent vulnerabilities, many of which may not be apparent to their commanders. TWIRL would be useful for identifying the critical functions and components of a combat force and hence their vulnerable features. This technique could be applied to identify enemy vulnerabilities for exploitation or to seek deficiencies in Blue for remedial action.

Red activities. By incorporating knowledge of Red field operations, doctrine, and tactics gained from observing exercises and actual combat, a composite, self-consistent picture of Red behavior could be built up over time. A knowledge-based simulation of the Red combat system (including air- and ground-based fire support, EC, maneuver unit control, firepower distribution and norms, logistics, air defense, sensor platform deployment, etc.) would help to provide insight into doctrine and procedures and to assess the individual and collective component contributions to Red combat capability.

Doctrine library. The TWIRL technique could support development of a computer database for Red and Blue doctrine, providing a single source for up-to-date, integrated Red combined arms operational doctrine, as well as NATO joint and combined doctrine, rules of engagement, procedures, norms, etc. Such a repository would enable analysts to compare differences between U.S. Service doctrine and U.S. joint doctrine, for example, and to identify and assess the effects of those differences.

Training. A comprehensive combat simulation could assist in teaching U.S. and NATO commanders how to make good decisions rapidly during combat by incorporating enemy doctrine, rules, limitations, and other behavioral patterns, as well as orders of battle and equipment capabilities. This tool could be used by candidate Blue commanders and their staffs to test their knowledge of current Red and Blue doctrines, their decisionmaking capabilities, and the operational procedures for the theater in scenarios that simulate combat situations.

Critical-node analysis. The TWIRL methodology could be used to help identify the nodes in enemy combat systems and to identify their characteristics when they are critical and when they are not. By analyzing simulated, potentially critical nodes at various times during the scenario (for example, when they are known to be critical), one could isolate the observable, characteristic behavior patterns associated with them, i.e., their discernible physical and electronic (COMINT, SIGINT, ELINT, and other) signatures. Field commanders need such data to determine which sensors and other combat information systems to deploy, where, and at which times.

Blue doctrine development. The TWIRL methodology could assist in developing and testing new Blue operational doctrine and in modifying current doctrine and procedures. It could also provide a means (now nonexistent) of harmonizing individual arms employment doctrine throughout the combined arms systems.

Evaluating contingency plans. Currently, contingency plans go largely untested, at least in their operational content. A TWIRL-like simulation could provide a tool for commanders' staffs to test developed plans, suggest deficiencies, identify potential successes, and help construct or modify plans to respond to new requirements.

This list is certainly not exhaustive, yet the applications described here are representative of the wide range of problem areas that need new tools to help in experimentation and analysis. The TWIRL development attempts to provide a step in that direction.

Appendix A

RIVER CROSSING OPERATION

The TWIRL simulation of a tactical combat operation highlights some of the EC activities of a complex military maneuver. Considerable material purporting to describe Soviet river crossing doctrine and tactics is available in the open literature (Department of the Army, 1977, 1978; Reznichenko, 1966; Sidorenko, 1970), but we did not attempt to include the entire range of functions and units described in that literature. Rather, we selected and aggregated them to simplify the simulation yet retain the elements that are essential to our purpose.

The simulation starts with the Red division in an assembly area about 70 km from the river. Units leave the assembly area, deploy for the approach march, and redeploy along the way into smaller functional units to arrive at their assigned assault positions. Preparatory fire and the subsequent assault order initiates the move to the river. The crossing is scheduled to begin at H-hour.

The planned movement schedule of each of the participating Red and Blue units is described in chronological order below. In this narrative description, we indicate only when units begin to move from a stationary position or when they halt after a move. (To completely define a track plan, however, the TWIRL code also contains unit speed and destination for each segment of a move.) Units are located according to an (x y) coordinate scheme that has the river located near an x value of 7.25 (for y values of interest here). (See Color Plate 10.) An exemplary Blue disruption effort was described and analyzed in Section VI, and the results were compared with the Red scheduled attack plan outlined below.

RED MOVEMENT SCHEDULE

H-7:00 Simulation begins.

H-6:50 DIVTAC, the Red division forward tactical headquarters, begins to move from the divisional assembly area (77.25 45.00) toward the start line on its approach march line of communication (LOC).

H-6:40 REGT1, the southern assault regiment, begins to move from the divisional assembly area (77.25 45.00) toward the start line on its approach march LOC.

H-6:30 REGT2, the northern assault regiment, begins to move from the divisional assembly area (77.25 45.00) toward the start line on its approach march LOC.

H-6:20 DIVTAC stops at (67.25 48.00) and opens its CP to monitor the move of the lead units to their approach march start lines.

DIVMAIN, representing the Red division's main headquarters and its remaining maneuver and support units, begins to move out of the assembly area (77.25 45.00) to establish a forward CP to take control when DIVTAC is on the move during the approach march.

H-6:10 REGTARTY11, one of two artillery battalions attached to and moving with REGT1, diverges to take up a position on its own approach march LOC from which to provide fire support to REGT1, if needed.

H-6:06 REGTARTY21, the second artillery battalion attached to REGT1, diverges to take up a position on its own approach march LOC from which to provide fire support to REGT1, if needed.

REGTARTY11 stops at (68.50 39.25) and sets up a temporary firing position.

H-6:05 REGT1 stops at (67.25 40.00) at the start line on its LOC.

H-6:02 REGTARTY21 stops at (68.50 40.75) and sets up a temporary firing position.

H-6:00 DIVARTY1, one of two artillery regiments subsumed in DIVMAIN, diverges to take up a position on its own approach march LOC from which to provide fire support to all divisional forces assigned to the southern (REGT1) attack axis.

H-5:55 REGT2 stops at (67.25 51.00) at the start line on its LOC.

DIVARTY2, the second artillery regiment subsumed in DIVMAIN, diverges to take up a position on its own approach march LOC from which to provide fire support to all divisional forces assigned to the northern (REGT2) attack axis.

H-5:50 DIVARTY1 stops at (68.50 42.50) and sets up a temporary firing position.

DIVMAIN stops at (67.25 45.00) and opens the divisional main CP.

H-5:45 DIVARTY2 stops at (68.50 49.00) and sets up a temporary firing position.

Color Plate 11 depicts a zoomed-in view of Red's forces at this point in the simulation.

H-5:30 DIVTAC departs from the start line (67.25 48.00) toward a forward location from which to monitor and control the marshaling of the assault units in their forward assembly areas.

REGT1 departs from its start line (67.25 40.00) on its approach march.

DIVARTY1 departs from its temporary firing position (68.50 42.50) on its approach march.

REGTARTY11 departs from its temporary firing position (68.50 39.25) on its approach march.

REGT2 departs from its start line (67.25 51.00) on its approach march.

DIVARTY2 departs from its temporary firing position (68.50 49.00) on its approach march.

H-5:00 DIVREAR begins to move from the divisional assembly area (84.00 44.00) toward a forward location from which to support the approach march of divisional units.

H-4:45 REGTARTY21 departs from its temporary firing position (68.50 40.75) on its approach march.

H−4:30 DIVTAC stops at (37.25 46.00), opens a CP, resumes control of the approach march, organizes, and instructs its accompanying signal intelligence and jamming units (DIVEV1, DIVEV2, ARMYEH, DIVJV1, DIVJV2, and ARMYJH) when to deploy.

DIVARTY1 stops at (38.50 42.00) and sets up a temporary firing position to support the approach march of units on the southern (REGT1) attack axis.

Color Plate 12 depicts a zoomed-in view of Red's forces at H−4:30.

H−4:24 DIVREAR stops at (67.25 42.00) and opens its rear services CP.

H−4:15 REGTARTY11 stops at (31.00 38.50) and sets up a temporary firing position to support the southern (REGT1) attack axis.

DIVMAIN departs from (67.25 45.00) toward a forward location from which to control the approach march when DIVTAC makes its next move into its final assault position.

H−4:03 DIVEV1 begins to deploy to an assault position on the left flank of REGT1.

DIVEV2 begins to deploy to an assault position on the right flank of REGT2.

ARMYEH begins to deploy to an assault position on the division axis to support both REGT1 and REGT2.

H−4:00 REGT1 closes its forward assembly area (22.25 40.00).

REGT2 closes its forward assembly area (22.25 50.00).

DIVARTY2 closes its forward assembly area (24.00 47.50).

ENG1 detaches from REGT1 and begins its move forward to survey crossing sites for the southern assault battalion of REGT1.

H−3:48 ARMYJH begins to deploy to an assault position on the division axis to support both REGT1 and REGT2.

DIVJV1 begins to deploy to an assault position on the left flank of REGT1.

DIVJV2 begins to deploy to an assault position on the right flank of REGT2.

H−3:45 ENG2 detaches from REGT1 and begins its move forward to survey crossing sites for the northern assault battalion of REGT1.

H−3:30 DIVMAIN stops at (44.75 46.70), opens its CP, and takes control of the approach march from DIVTAC (which prepares to deploy forward to a position from which to control the assault).

DIVARTY1 departs from its temporary firing position (38.50 42.00) for its forward assembly area.

Color Plate 13 depicts Red's forces at H−3:30.

H−3:15 REGTARTY21 closes its forward assembly area (23.50 40.74).

H−3:00 DIVARTY1 closes its forward assembly area (24.00 42.50).

DIVTAC closes its CP (37.25 46.00) and departs for its forward position from which to control the assault.

H−2:45 REGTARTY11 departs from its temporary firing position (31.00 38.50) for its forward assembly area.

H−2:30 DIVTAC closes its final assault position (24.00 45.00), opens its CP, and resumes control.

FWD111, the assault company of the southern lead battalion of REGT1, departs from the regimental forward assembly area for its assault position.

ADV11, the southern assault battalion of REGT1, departs from the regimental forward assembly area for its assault position.

FWD121, the assault company of the northern lead battalion of REGT1, departs from the regimental forward assembly area for its assault position.

ADV21, the northern assault battalion of REGT1, departs from the regimental forward assembly area for its assault position.

REGTARTY11 closes its forward assembly area (23.50 39.25).

BNARTY211, an artillery battery detached from REGTARTY11, departs from its forward assembly area for its assault position.

BNARTY121, an artillery battery detached from REGTARTY21, departs from its forward assembly area for its assault position.

H−2:15 REGTARTY21 departs from its forward assembly area for its assault position.

H−2:00 DIVMAIN closes its CP and departs for its assault position.

DIVARTY1 departs from its forward assembly area for its assault position.

BNARTY121 and BNARTY211 close their assault positions ((12.75 42.00) and (12.75 38.00), respectively) and report ready to assault.

FWD111, FWD121, ADV11, and ADV21 close their assault positions ((12.25 38.50), (12.25 41.50), (12.25 37.00), and (12.25 43.00), respectively) and report ready to assault.

FWD112, the assault company of the southern assault battalion of REGT2, departs from the regimental forward assembly area for its assault position.

ADV12, the southern assault battalion of REGT2, departs from the regimental forward assembly area for its assault position.

FWD122, the assault company of the northern assault battalion of REGT2, departs from the regimental forward assembly area for its assault position.

ADV22, the northern assault battalion of REGT2, departs from the regimental forward assembly area for its assault position.

H−1:53 ARMYJH closes its assault position (18.00 44.00) and reports operational.

H−1:45 REGTARTY21 closes its assault firing position (13.50 40.74) and reports ready to commence preparatory fires.

BNARTY221, an artillery battery detached from REGTARTY21, departs from its forward assembly area for its assault position.

BNARTY111, an artillery battery detached from REGTARTY11, departs from its forward assembly area for its assault position.

H−1:39 ARMYEH closes its assault position (18.27 45.00) and reports operational.

H−1:30 REGTARTY11 departs from its forward assembly area for its assault position.

DIVARTY1 closes its assault firing position (19.00 42.50) and reports ready to commence preparatory fires.

ENG1 completes its survey, halts, and establishes a crossing central point for FWD111 and ADV11.

DIVMAIN closes its assault position (29.75 45.43).

FWD112, FWD122, ADV12, and ADV22 close their assault positions ((12.25 48.50), (12.25 51.50), (12.25 47.00), and (12.25 53.00) respectively), and report ready to assault.

H−1:15 DIVREAR starts to move from (67.25 42.00) for its forward position from which to support the assault units.

DIVARTY2 departs from its forward assembly area for its assault position.

BNARTY111 and BNARTY221 close their assault positions ((12.75 36.00) and (12.75 44.00), respectively) and report ready to assault.

ENG2 completes its survey, halts, and establishes a crossing control point for FWD121 and ADV21.

H−1:03 DIVJV1 and DIVJV2 close their assault positions ((10.00 36.50) and (10.00 53.50), respectively) and report operational.

DIVEV1 and DIVEV2 close their assault positions ((10.25 37.50) and (10.25 52.50), respectively) and report operational.

H−1:00 REGTARTY11 closes its assault position (13.50 39.25) and reports ready to commence preparatory fires.

Color Plate 14 depicts Red's (and Blue's) forces at H−1:00.

H−0:45 REGT1 issues the assault order: jamming to begin immediately, the preparatory fires to begin at H−0:30, and the assault units to jump off at H−0:15.

DIVJV1 and ARMYJH begin jamming.

DIVARTY2 closes its assault position (19.00 47.50) and reports ready to commence preparatory fires.

H−0:30 REGTARTY11 opens preparatory barrage on forward Blue defensive positions covering crossing points for FWD111 and ADV11.

REGTARTY21 opens preparatory barrage on forward Blue defensive positions covering crossing points for FWD121 and ADV21.

DIVARTY1 opens preparatory barrage on second-line Blue defensive positions across the entire REGT1 frontage.

H−0:15 DIVREAR closes its assault position (38.50 42.00).

FWD111, FWD121, FWD112, and FWD122 begin move from assault positions to crossing sites.

BNARTY111, BNARTY211, BNARTY121, and BNARTY221 begin move from assault positions to assault firing positions.

H−0:10 BNARTY111, BNARTY211, BNARTY121, and BNARTY221 stop and open direct fire on the Blue forward positions covering the crossing sites.

H−0:00 FWD111, FWD121, FWD112, and FWD122 reach the river and begin to cross.

BNARTY111 and BNARTY221 cease fire and begin to move toward their crossing sites.

ADV11, ADV21, ADV12, and ADV22 begin to move from assault positions to their crossing sites.

REGT1 and REGT2 begin to move from assault positions to their crossing sites.

Color Plate 15 depicts Red's (and Blue's) forces at H-hour.

H+0:15 BNARTY121 and BNARTY211 cease preparatory barrage on Blue forward defense positions and revert to on-call missions.[1]

FWD111, FWD121, FWD112, and FWD122 reach opposite shore and begin advance on immediate objective areas (i.e., Blue forward defense positions).

ADV11, ADV21, ADV12, and ADV22 close the river and begin to cross.

BNARTY111 and BNARTY221 close the river and begin to cross.

H+0:30 DIVARTY1 and REGTARTY21 cease preparatory barrage fire and revert to on-call missions.

REGTARTY11 ceases preparatory barrage fire and begins displacement to new firing position closer to the river.

ADV11, ADV21, ADV12, and ADV22 reach the opposite shore and advance on the Blue forward defense positions.

FWD111, FWD121, FWD112, and FWD122 reach Blue forward defense positions and engage.

BNARTY111 and BNARTY221 reach opposite the shore and move toward advanced firing positions to support assault on Blue forward positions.

DIVARTY2 begins displacement to a new firing position closer to the river.

[1] Behaviors to implement on-call artillery missions for Red have not been included in the present version of TWIRL. This could be done through a process similar to that described for Blue (see Section IV), although the decision rules would probably be different.

Color Plate 16 depicts Red's (and Blue's) forces at H+0:30.

H+0:45 BNARTY111 and BNARTY221 set up new advanced firing positions ((6.50 36.00) and (6.50 44.00), respectively).

BNARTY121 and BNARTY211 begin to move to crossing sites.

REGTARTY11 sets up at a new firing position (10.25 39.25).

REGTARTY21 begins move to a new firing position closer to the river.

ADV11, ADV21, ADV12, and ADV22 reach Blue forward defense positions and reinforce FWD assault units engaging the defenders.

REGT1 and REGT2 close the river and begin to cross.

H+1:00 REGT1 and REGT2 reach the opposite shore and begin advance in support of FWD/ADV units.

BNARTY121 and BNARTY211 close the river and begin to cross.

REGTARTY21 sets up at a new firing position (10.25 40.75).

REGTARTY11 begins to move to crossing site.

DIVARTY2 sets up at a new firing position (14.00 47.50).

DIVARTY1 begins move to a new firing position closer to the river.

H+1:08 FWD111, FWD121, FWD112, and FWD122 penetrate Blue forward defense positions and advance on final objective areas.

H+1:15 ADV11, ADV21, ADV12, and ADV22 secure intermediate objective areas and advance on final objectives.

DIVTAC turns over control of the operation to DIVMAIN and begins to move to crossing site.

REGTARTY11 closes the river and begins to cross.

BNARTY121 and BNARTY211 reach the opposite shore and move toward advanced firing positions.

H+1:30 REGTARTY11 reaches the opposite shore and moves toward an advanced firing position.

BNARTY121 and BNARTY211 set up at advanced firing positions ((5.00 42.00) and (5.00 38.00), respectively).

BNARTY111 and BNARTY221 begin to move to firing positions for assault on final objective areas.

H+1:38 FWD111, FWD121, FWD112, and FWD122 reach final objective areas and engage defenders.

H+1:45 ADV11, ADV21, ADV12, and ADV22 reach final objective areas and reinforce FWD assault units.

DIVARTY1 sets up at a new firing position (10.25 42.50).

DIVARTY2 begins to move to crossing site.

REGTARTY11 sets up at its final advanced firing position (5.50 39.25).

REGTARTY21 begins to move to crossing site.

BNARTY111 and BNARTY221 set up at final firing positions ((5.00 36.00) and (5.00 44.00), respectively).

H+2:00 REGT1 and REGT2 reach final objective areas and join in the assault on Blue defenders.

REGTARTY21 closes the river and begins to cross.

H+2:08 DIVTAC closes the river and begins to cross.

H+2:15 DIVARTY2 closes the river and begins to cross.

REGTARTY21 reaches the opposite shore and moves toward its final firing position.

H+2:23 DIVTAC reaches the opposite shore and moves out to establish a forward CP for the final assault.

H+2:30 DIVARTY2 reaches the opposite shore and moves toward its final firing position.

REGTARTY21 sets up at its final firing position (5.50 40.75).

H+2:38 DIVTAC opens a forward CP at (6.00 45.00) and resumes control of assault forces from DIVMAIN.

H+2:45 DIVARTY1 begins to move to crossing site.

DIVARTY2 sets up its final firing position (6.00 47.50).

H+3:00 DIVMAIN begins to move forward to establish a CP to control crossing by second-echelon regiments and supporting units.

All FWD/ADV/REGT units halt with their final objective areas secured. Final positions are FWD111 (0.38 38.50), FWD121 (0.38 41.50), FWD112 (0.38 48.50), FWD122 (0.38 51.50), ADV11 (0.50 37.00), ADV21 (0.50 43.00), ADV12 (0.50 47.00), ADV22 (0.50 53.00), REGT1 (1.00 40.00), REGT2 (1.00 50.00).

DIVARTY1 closes the river and begins to cross.

H+3:15 DIVARTY1 reaches the opposite shore and moves toward its final firing position.

H+3:30 DIVMAIN halts at (14.75 45.00) and opens a CP.

DIVARTY1 sets up at its final firing position (6.00 42.50).

BLUE MOVEMENT SCHEDULE

The simulation starts with the defending Blue mechanized infantry regiment in an assembly area about 70 km west of the river line. When ordered, deployment forward will be by battalion over a single LOC leading to a series of ridge lines overlooking the river. Times listed are on the same time line as Red, i.e., simulation begins at H−7:00 and Red plans to have its first units crossing the river at H-hour.

H−6:30 Blue AIRMISSION1 takes off from the airbase at (−150.00 60.00) to reconnoiter Red approach routes to potential river crossing sites.

H−6:13 Blue AIRMISSION1 arrives in its assigned surveillance area (70.00 60.00).

H−6:04 Blue AIRMISSION1 departs from its assigned surveillance area at (30.00 30.00).

H−5:50 Blue AIRMISSION1 lands at the airbase (−150.00 60.00).

H−5:15 MECHBN1 departs from the assembly area (−60.00 43.00) for its river defense position.

H−5:00 MECH-REGT-HQ1 departs from the assembly area (−60.00 43.00).

Blue AIRMISSION2 takes off from the airbase at (−150.00 60.00) to continue route surveillance.

H−4:45 MECHBN2 departs from the assembly area (−60.00 43.00).

H−4:43 Blue AIRMISSION2 arrives in its assigned surveillance area (70.00 60.00).

H−4:34 Blue AIRMISSION2 departs from its assigned surveillance area at (30.00 30.00).

H−4:30 MECHBN3 departs from the assembly area (−60.00 43.00).

H−4:20 Blue AIRMISSION2 lands at the airbase (−150.00 60.00).

H−4:15 MECHBN4 departs from the assembly area (−60.00 43.00).

H−3:30 Blue AIRMISSION3 takes off from the airbase at (−150.00 60.00) to continue surveillance.

H−3:13 Blue AIRMISSION3 arrives in its assigned surveillance area (70.00 60.00).

H−3:04 Blue AIRMISSION3 departs from the assigned surveillance area at (30.00 30.00).

H−2:50 Blue AIRMISSIUN3 lands at the airbase (−150.00 60.00).

H−2:48 JAMCO1, marching with MECH-REGT-HQ1, detaches and moves toward its defensive operating site.

H−2:42 ARTYBN1, marching with MECH-REGT-HQ1, detaches and moves toward its CP location and the firing position for its attached gun battery.

H−2:37 ARTYBTRY1, marching with MECHBN1, detaches and moves toward its firing position.

H−2:28 MECH-REGT-HQ1 reaches its defensive position (1.25 46.50) and opens its CP.

MECHBN1 reaches its defensive position (5.50 40.50) and digs in.

H−2:27 JAMCO1 reaches its assigned operating location (1.50 43.50).

H−2:24 ARTYBN1 reaches its defensive position (1.00 48.00), opens its artillery CP, and sets up its attached gun battery.

H−2:21 DFPLT1, marching with MECHBN3, detaches and moves toward its operating site.

H−2:20 ARTYBTRY1 reaches its assigned firing position (5.50 38.50) and sets up.

H−2:07 ARTYBTRY2, marching with MECHBN2, detaches and moves toward its firing position.

H−2:06 MECHBN3 reaches its defensive position (1.50 41.00) and digs in.

H−2:02 DFPLT1 reaches its assigned operating site (1.50 39.00).

H−1:58 MECHBN2 reaches its defensive position (5.50 49.50) and digs in.

H−1:50 ARTYBTRY2 reaches its assigned firing position (5.50 51.50) and sets up.

H−1:45 DFPLT2, marching with MECHBN4, detaches and moves toward its operating site.

H−1:30 MECHBN4 reaches its defensive position (1.50 49.50) and digs in.

H−1:28 DFPLT2 reaches its assigned operating site (1.50 51.00).

The final disposition of Blue's forces can be seen in Color Plate 14.

Appendix B

COMPUTING ENVIRONMENT

ROSS has been operational for several years and has been implemented in a wide variety of Lisps (Narain et al., 1983), including Maclisp, Franzlisp, Interlisp-20, Vax-Interlisp, Interlisp-D, and Zetalisp. Our primary research environment, and the one in which TWIRL is implemented, includes the Franzlisp version of ROSS running under UNIX 4.1bsd on either a VAX 11/780 or VAX 11/750.

A file containing Franzlisp (Opus 38.50, dated February 1983) requires 605,302 bytes of storage. Adding compiled ROSS requires an additional 146,432 bytes. TWIRL contains approximately 140 objects and 200 behaviors. A system including Franzlisp, ROSS, and compiled TWIRL code requires 1,203,318 bytes of storage.

Our graphics software is written in the C language and loaded directly into the Franzlisp environment. Our primary graphics processor is an AED 512 with a color monitor. We have interfaced ROSS directly with an EMACS editor. Rand's E editor can also be used to edit ROSS files, but in using E, editing must be performed external to the ROSS environment.

BIBLIOGRAPHY

Conker, R. S., J. R. Davidson, P. K. Groveston, and R. O. Nugent, *The Battlefield Environment Model*, The Mitre Corporation, MTR-83W00245, September 1983.

Department of the Army, *Opposing Forces—Europe*, Field Manual FM 30-102, Washington, D.C., 18 November 1977.

Department of the Army Intelligence and Security Command, *Soviet Army Operations*, U.S. Army Intelligence and Threat Analysis Center, JAG-13-U-78, April 1978.

Dockery, J. T., *Structure of Command and Control Analysis*, Shape Technical Center, The Netherlands, 1982; presented at the Symposium on Modeling and Analysis of Defense Processes, Brussels, July 1982.

Gunsch, G. H., and B. S. Hebert, *A Proposed Military Planning Task Simulator Using the ROSS Language*, Masters Thesis, Air Force Institute of Technology, AFIT/GE/EE/83D-24, 1983.

Jefferson, D., and H. Sowizral, *Fast Concurrent Simulation Using the Time Warp Mechanism, Part I: Local Control*, The Rand Corporation, N-1906-AF, December 1982.

Klahr, P., and W. S. Faught, "Knowledge-Based Simulation," *Proceedings of the First Annual National Conference on Artificial Intelligence*, Palo Alto, 1980.

Klahr, P., D. McArthur, and S. Narain, "SWIRL: An Object-Oriented Air Battle Simulator," *Proceedings of the Second Annual National Conference on Artificial Intelligence*, Pittsburgh, 1982a, pp. 331–334.

Klahr, P., D. McArthur, S. Narain, and E. Best, *SWIRL: Simulating Warfare in the ROSS Language*,, The Rand Corporation, N-1885-AF, September 1982b.

McArthur, D., and P. Klahr, *The ROSS Language Manual*, The Rand Corporation, N-1854-AF, September 1982.

McArthur, D., P. Klahr, and S. Narain, *ROSS: An Object-Oriented Language for Constructing Simulations*, The Rand Corporation, R-3160-AF (forthcoming).

Narain, S., D. McArthur, and P. Klahr, "Large-Scale System Development in Several Lisp Environments," *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, 1983, pp. 859–861.

Nugent, R. O., "A Preliminary Evaluation of Object-Oriented Programming for Ground Combat Modeling," The Mitre Corporation, Working Paper WP-83W00407, 27 September 1983.

Reznichenko, V. G., *Tactics (The Officer's Library)*, Moscow, 1966, translated by USAF Foreign Technology Division, FTD-MT-67-35, NTIS, AD659928, 1967.

Sidorenko, A. A., *The Offensive (A Soviet View)*, Moscow, 1970, translated and published by U.S. Air Force, Washington, D.C., 1975.

Steeb, R., D. McArthur, S. Cammarata, S. Narain, and W. Giarla, *Distributed Problem Solving for Air Fleet Control: Framework and Implementation*, The Rand Corporation, N-2139-ARPA, April 1984.

RAND/R-3158-AF