LOAN DOCUMENT

PHOTOGRAPH THIS SHEET

DTIC ACCESSION NUMBER

LEVEL

INVENTORY

D

NMD BM/C3 Implementation Sep 96

DOCUMENT IDENTIFICATION

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

DISTRIBUTION STATEMENT

HANDLE WITH CARE

ACCESSION FOR

NTIS	GRA&I	☑
DTIC	TRAC	☐
UNANNOUNCED		☐
JUSTIFICATION		

BY

DISTRIBUTION/

AVAILABILITY CODES

DISTRIBUTION	AVAILABILITY AND/OR SPECIAL
A-1	

DISTRIBUTION STAMP

DATE ACCESSIONED

19970114 010

DATE RETURNED

DATE RECEIVED IN DTIC

REGISTERED OR CERTIFIED NUMBER

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

DTIC FORM 70A JUN 90

DOCUMENT PROCESSING SHEET

PREVIOUS EDITIONS MAY BE USED UNTIL STOCK IS EXHAUSTED.

LOAN DOCUMENT

17 JAN 1997

NMD BM/C3 Implementation

Robert Swenson

PEO Missile Defense
National Missile Defense
Battle Management
Division

Huntsville, Alabama

Abstract

In August of 1995, BMDO awarded a contract to TRW to develop a rapid prototype BMC3 systems for National Missile Defense. This was a streamlined acquisition effort, focusing on reduced CDRLs, paperless system, and joint Government - contractor Integrated Product Teams (IPT). The leader of this effort was LTC Jim McKenna, now retired from the Air Force. The acquisition process has been publicly presented (see "Contract Management", February 1996 issue). This paper is a follow on to that article and briefly describes the BMC3 product and two of the key TRW - Government IPT processes used to date in this streamlined development effort.

Overall Program Schedule

The basic program covers four years and requires the development of a Battle Management, Command, Control, and Communication (BMC3) prototype. The contract includes a three year option for a potential total contract length of seven years. There are seven Capability Increments spread over those seven years. The first increment, Capability Increment 1 (CI 1), is just concluding. This increment established the basic system architecture, defined the development environment, and established the development processes. An initial and basic set of BMC3 capability, mostly from re-use software, was developed for the Mission Application Software (MAS) within the BMC3. Each succeeding increment will increase the robustness and maturity of the MAS as the program evolves. The product of each Capability Increment represents a complete BMC3, in that all the basic subsystems (battle planning, human in control, displays, etc.) are there at ever increasing levels of maturity. These products are then used to support integrated ground tests (IGT), integrated flight test (IFT), and user assessment. Figure 1 illustrates the schedule for the increments in the first four years (basic contract). An initial set of BMC3 requirements was developed and presented at the Initial BMC3 Requirements Review, in fourth quarter CY95. These requirements were then rolled into the capability Increment 1 software development effort. An Initial Review (IR) was conducted in first quarter CY96 which resulted in the baselining of the requirements for CI 1. The Release Review for CI 1 (not yet conducted at the time of this

writing) will mark the end of the development phase for CI 1. At the RR, the BMC3 product will be ready for use in IGTs, IFTs, and User Assessment.

Figure 2 takes one Capability Increment build sequence and provides further detail as to the sequence of events and relationship between IR, RR, and other development events. The milestone reviews are the initial BMC3 review, the IR, and the RR. The IR provides the baseline requirements to the development process where the requirements are further refined and derived requirements are identified. Actor and data classes are developed to implement these requirements. The results of this process is an overarching design for this increment. At this point, a Design Walk Through (DWT) is conducted to scrub the design for defects. After the DWT, the actual coding, or object class implementation, begins. At the same time, steps are taken to prepare for integrating the various objects and actor classes. BMC3 test planning also begins. As individual actor classes reach maturity for that increment, Code Walk Throughs (CWT) are conducted to identify and remove defects. As the now coded object classes are completed, software integration activities of these objects are done. Although figure 2 may imply a waterfall, sequential process, it is, in fact, an iterative process between coding and integration as the Capability Increment takes shape. As the software integration reaches conclusion, a TWT is scheduled and conducted. This is done to verify the product, and its associated test procedures and scenarios are ready for release testing. The BMC3 product now enters release testing where requirement verification takes place. Upon successful completion of release testing, a final review, RR, is conducted to present the product of this Capability Increment to the community.

The BMC3 Products

The BMC3 is a Real Time Object Oriented Methodology (ROOM) design with five major actor container classes and communication services. Figure 3 illustrates the hardware configuration for a single BMC3 node and identifies the five major actor classes.

Figure 4 describes the major functions of each of the five major actor container classes. The BMC3 is being developed as a set of common software, capable of being configured as a Commander in Chief (CINC) or site node. The former would be applicable for a CINC BMC3 at a national command center, the latter as a Site BMC3 at an interceptor base or other defensive element location.

The software is implemented in Ada 95 and utilizes an Integrated Engineering Infrastructure (IEI) as shown on Figure 5.

The BMC3 development is augmented by the development of a test and evaluation tool, Test Exerciser (TEx). TEx will be used as a BMC3 system driver / test tool,

providing test control, NMD weapon and sensor models, and threat models / drivers. The NMD system elements modeled by TEx for Capability Increment #1, are illustrated in Figure 6. The Defense Support Program (DSP) and Early Warning Radar (EWR) models are notional representations of national sensor assets. Additional models include the Early Warning System (EWS), the Ground Based Radar (GBR), the Ground Based Interceptors (GBI) and their associated launch facilities (GBI Farm), and the In Flight Interceptor Communications System (IFICS). Two nodes of BMC3 - one representing a CINC and one representing a site - interact to form a BMC3 system. The DSP and EWR provide simulated early warning alerts of threatening launches. The outputs of these sensor models are consolidated in the Early Warning System (EWS) model and then passed on to the CINC BMC3. These alerts initiate a defensive response from the BMC3 system. The BMC3 system then tasks the defensive elements, such as the GBR, GBI, and IFICS, to engage and negate the simulated attack. For CI 1, all sensor models are relatively simple. Threats are flown as truth data. Perceived data from the sensors is truth data with the addition of random noise to simulate real data.

TEx is also capable of simulating a launch / intercept sequence representative of a Kwajalein Missile Range (KMR) flight test. This allows the BMC3 nodes to be exercised in a flight test scenario as well as a tactical one.

Process for Software Development

Figure 7 illustrates the software development flow. A Government provided Information Architecture (IA) is used as the input for the ObjecTime modeling tool. From the IA, a Software Architecture Skeleton (SAS) is built using actor classes, protocols, port definitions, and transitions in ObjecTime. This SAS is a static structure of all messages, both internal and external to the BMC3. The SAS defines the interconnectivity between actor classes, their port names, and protocol names. The SAS is generated from ObjecTime as a computer file called Linear Form. This Linear Form file is fed into the Process Construction System (PCS), part of the IEI. From the Linear Form input, PCS auto generates an Ada 95 source code representation of the SAS. This consists of Ada specifications and stubs. The original ObjecTime messages are now in Ada spec's, the ObjecTime ports are UNIX sockets, and the ObjecTime protocols are UNAS bindings. The functionality of each BMC3 stub is fleshed out in the Mission Application Software (MAS) in increasing levels of capability and robustness with each capability build. This MAS will consist of reused code and new, "hand written" code. The MAS and SAS are ultimately linked together to form the BMC3 executable code.

We will now turn our attention to some of the processes used in the BMC3 development.

Pacing BenchMarks (PBM)

Pacing BenchMarks are used to monitor technical and schedule progress. The objective of Pacing Bench Marks is to provide demonstration based progress at scheduled intervals. It does not take the place of contractual milestones, release testing, or formal demonstrations. Its relationship to Contractual Milestones and the characteristics of both are described Table 1.

Key things to note are that Pacing BenchMarks are informal engineering sessions that require no further effort on the part of the developer than that which they would normally have to do to develop the product. Pacing BenchMarks are just a tool to assure that product development is pacing itself smoothly along toward the Contractual Milestones. They demonstrate that the objectives of the Pacing BenchMark have been achieved. This is done by real time on the spot execution of the subject software, followed by on the spot assessment by cognizant Government engineers. All this is done on the day of the Pacing BenchMark. The following paragraphs provide a more detailed description of this process:

After the requirements for each Capability Increment build have been baselined at the IR, the Pacing BenchMark schedule and PBM objectives for the current Capability Increment are identified in a joint TRW / Government planning session. The two guiding principles for this session are:

1) Plan to succeed.
2) Pacing BenchMarks track development, but don't drive it.

This planning session is at the engineering level and involves the TRW Software Development Lead, the TRW Capability Increment lead, and the executing service lead or point of contact. Additional TRW engineering support is brought in on an "as needed" basis. This group lays out a schedule of Pacing BenchMarks and the objectives of each. One of the main responsibilities of the group is to realistically size the job, taking into account the resources available (i.e., staff, skill mix, time, development assets available, dollars) and the Capability Increment requirements. The "Plan to Succeed" principle is applied by assuring that the resources are available to meet the Pacing BenchMarks which in turn support meeting the overall Capability Increment requirements. Software estimating tools, SLOC estimates, engineering judgment all play into this. The second principle, "Pacing BenchMarks track development, but don't drive it" is also applied. The objective here is to assure that the Pacing BenchMarks and their objectives merely reflect clean and logical "check points" in the development process that the design engineers would follow even in the absence of Pacing

BenchMarks. The rule of thumb is that there should be no extra development effort just to "pull off" a Pacing BenchMark ... just the normal engineering activity that needs to be done to have the product developed to the point it should be by the time of the Pacing BenchMark. This includes training the Government engineers. It is incumbent upon the Government engineers to stay abreast of technical development through active participation in the Integrated Product Teams (IPT).

The products that come out of this planning session are (1) a schedule of Pacing BenchMarks, coordinated with programmatic and contractual milestones and (2) the technical objectives of each Pacing BenchMark. Generally, Pacing BenchMarks occur about every two months

With the Capability Increment requirements defined, the Pacing BenchMarks scheduled, resources assessed, and objectives defined, the detailed check list for the next Pacing BenchMark is developed. This is an iterative process of strawman development, discussions at technical interchange meeting, individual review, and strawman change. The objective here is to achieve a detailed check list that is exact, measurable, and clearly demonstrates attainment of the Pacing BenchMark objectives. All issues of scenarios, software execution, computer simulations, measurement of required data, logging, etc. are addressed at this time. At least one real time computer execution is required on the day of the Pacing BenchMark. Additional runs and / or prior runs are acceptable. However, for the latter, proper configuration controls must be in place to assure any prior runs are with the same software baselined for the Pacing BenchMark. These are informal engineering runs; no formal QA support is required. For convenience, such formal QA is allowed if, in the judgment of the developer, it will facilitate baselining the subject software.

On the day of the Pacing BenchMark, cognizant Government engineers will arrive to conduct the PBM. Cognizant Government engineers means that these are individuals who have been involved in monitoring the technical development of the subject software by active participation in the IPTs, are knowledgeable of the technical aspects of the subject software, and are competent to render judgment using the PBM Check List. No other Government representation is allowed. The subject software is baselined for the PBM. The agreed upon software execution is conducted by TRW personnel and witnessed by the Government engineers. The results of this run - and any prior agreed upon runs - are then reviewed by the Government engineers. The coder for each software module is present to assist in interpreting displays, answering questions, etc. The Government engineer compares the results with the requirement in the PBM List and makes a pass / fail assessment. This is repeated for each item in the check list. Usually

multiple assessments of different parts of the subject software are going on at the same time, involving several pairs of Government and developer engineers. At the end of this assessment period, a Government only caucus is held in which a final PBM score is tallied. This score is the percentage of the number of check list items successfully passed. Any failed items are evaluated as to their criticality. Each Government engineer maintains his / her own annotated check list for those portions of the subject software they assessed. These check lists are collected during the Government only caucus and kept as a permanent record. The Government engineers arrive at an overall PBM assessment, in addition to the total score. This overall assessment can take the form of one of four possible outcomes outlined in Table 2.

These overall assessments are the technical recommendations to Government management who then has the final disposition authority. Depending on the outcome of the BenchMark, one of the actions described in Table 3 occur.

Figures 8 and 9 illustrate an actual Pacing BenchMark. In this example, the objective was to demonstrate that software development environment, from ObjecTime to executable code was operating and that the process was repeatable. It also demonstrated that the ObjecTime model was built and the "first cut" SAS was completed, for that iteration. Two computer runs were made for this PBM. The first run, the control run, consisted of the baselined SAS demo with executable Ada code, run at the contractors facility. The event traces of the ObjecTime model were verified, the ObjecTime Linear Form output was compared against the ObjecTime models, the Ada source code was checked against the ObjecTime models, and the event traces from actual run were verified. Next, the entire software process was repeated, with Government evaluators as witnesses, as a Test Case Run. ObjecTime generated another Linear Form, still from the models baselined for the Control Run; the Linear Form was run through the PCS to generate another set of Ada spec's. The Ada spec's were run through the compiler (GNATT), producing an object code version of the SAS. The SAS object code was re-linked with the MAS code from the control run. The SAS demo was repeated with this new set of executable SAS code and the event traces again verified. In addition, the Linear Form and selected files from the Ada source code were electronically compared between the Control Run and the Test Case run, to verify repeatability. This PBM was executed at the TRW facilities in Huntsville, Alabama, on Thursday, May 10, 1996. The overall score was an 85 and a Conditional Pass was awarded, with defects to be resolved and re-run at the next PBM. This concludes our discussion of the Pacing BenchMark process. Another key Software Development process, described in the following paragraphs, was Defects Removal.

Defects Removal

The objective of the Defects Removal Program (DRP) was to eliminate defects in software before delivery to the customer. Its relationship to Release Testing is described below.

A full Defects Removal Program is costly. For example, on a previous Army Rapid Prototype development effort, defects removal, including documentation and tracking, took 26.3% of the total development labor hours expended. The BMC3 RFP did not require a full DRP nor did TRW bid one. However, the requirement to launch missiles at KMR as part of the IFT's and the potential for deployment under a contingency deployment made it prudent for the Government and TRW to take all measures possible to achieve quality and robust software under the BMC3 contract as it stood. The DRP described in this paper represents a compromise between the realities of the current contract and a full DRP. Design Walk Throughs (DWT) were conducted on each of the major actor classes (C2, Threat States, Tasker, Planner, System States) and Tex. Code Walk Throughs (CWT) were conducted under the following guidelines: (1) each coder was to go through at least one CWT, (2) each major actor class was to go through at least one CWT, (3) any critical modules, as designated by the TRW Chief Engineer, were to go through a CWT. To further prioritize the problem, the candidates for CWTs were limited to new, hand written code. Such code was deemed as the most probable source of potential defects. CWTs were conducted on the Source Tracker module (part of Threat States), the Evaluated Object module (part of Threat States), various sub-systems of the Planner, System Tracker (part of Threat States), the Task Plan Protocol module (primarily affects Planner), Element Tasker, the Resource State Server Client module (part of System States), and the Generic Client - Server module (used through out the BMC3). In future capability increments, the use of CASE tools such as AdaMat, Ada Analyzer, and TestMate will support this effort and assist by automating, to the extent possible, the defects discovery effort, as well as providing test coverage metrics. These tools were not available during Capability Increment #1. DWTs and CWTs will still be continued in future increments as well.

Formal peer reviews were conducted for designs and code. These reviews were not information briefings for management or general consumption. The objective of the review was to raise technical issues associated with the product under review. The reviews did not attempt to solve the issues so raised. The amount of material reviewed was limited to that which could be done in a two hour period ... for example, for code, this equated to about 300 SLOC (one SLOC = code between semicolons). The review was held when the author indicates his/her work was ready for review. At this time, the review was scheduled by the TRW chief engineer. Prior to the

review, the author prepared a review package and distributed it to the designated peer reviewers, who were chosen from the author's peers and designated in advance by the chief engineer. The reviewers spent at least two hours reading and commenting on the review package prior to the scheduled review. The actual review was conducted by a team of 5-7 people. A hard and fast rule that was enforced was that no one attended who was not scheduled to be there. Those selected were technically qualified to contribute, prepared to do so, and trained in the review process. The only exception was some initial monitoring by an IV&V representative. The review team consisted of a moderator, the author (who also served as the recorder), and 2 - 4 reviewers. The role of the moderator was to assure a good review or know the reason why. The role of the author was to provide technical clarification and to record issues raised. The role of the reviewers was to raise issues only, not solve them. Guidelines for selecting reviewers included: (1) all reviewers should be technically qualified to contribute to the review; (2) no reviewers should have any conflicts of interest (e.g., reviewing your own product); (3) all reviewers should also have participated as an author in another review (i.e., true peer reviews ... all reviewers should also be reviewed). All participants - moderator, author, reviewer, shall be trained in the review process. Three products came out of the review: a Summary Sheet, an Issues List, and a Related Issues List. The Summary Sheet, filled out by the moderator, documented the date of the review and the consensus findings of the reviewers. The reviewers had four choices for their consensus finding: (1) accept the review package as is, (2) accept with minor revisions (some changes required but of such a magnitude as to not require another review), (3) major revisions (many changes required and of such a magnitude as to require another review), (4) scrap and rework. Consensus is required ... if *one* reviewer felt strongly that the review package was not "good to go", the Summary Sheet would reflect that position. The reviewers attested to the consensus finding by signing the Summary Sheet. The Issues List was just that ... a list of issues, concerning the review package, that were raised during the review. The Related Issues List contained issues raised that were not a direct part of the review package being reviewed but that affected other modules. These three products were copied and distributed to all affected parties - particularly those affected by the Related Issues List - at the end of the review. The original Summary Sheet was retained by the Government. Ultimately, a tally (metric) was kept of all defects removed.

After the review, the tracking, resolution, and close out phase was entered. This started with the TRW Chief Engineer and the author reviewing the issues lists, identifying and removing non-problems and redundant issues and determining severity. The remaining issues were then designated defects and were

cataloged and tracked until closed out. Severity - which represented the consequence of the defect remaining - were characterized on a scale of 1 - 4. (1- crash, 2- interruption to major function, 3- interruption to minor function, 4- superfluous). The Summary Sheet original was kept by the Government along with a copy of the Related Issues List and Issues list. Prior to the next PBM, the Government verified that all the defects, at least those which affected that PBM, were corrected. Once all issues were corrected, the Government destroys all Related Issues List and Issues List. *... only the Summary Sheets and a tally of defects removed are kept.* . The Defects Tally was kept by main container actor class, by design phase, and by severity. Periodic reports of open and closed issues were provided at each Build Team Meeting. These meetings were held on a monthly basis and were scheduled by the Government Capability Increment Lead. For Capability Increment #1, this was Capt. Jeff Blank, Electronic Systems Command, Hanscom Air Force Base, Mass.

As mentioned, Design Walk Throughs (DWT) were conducted on each of the major actor classes (C2, Threat States, Tasker, Planner, System States) and Tex. Code Walk Throughs (CWT) were conducted on the Source Tracker module (part of Threat States), the Evaluated Object module (part of Threat States), various sub-systems of the Planner, System Tracker (part of Threat States), the Task Plan Protocol module (primarily affects Planner), Element Tasker, the Resource State Server Client module (part of System States), and the Generic Client - Server module (used through out the BMC3). Figure 10 summarizes the results of the DWTs and CWTs.

Conclusion

This concludes our look at the BMC3 product and some to the key IPT processes used. It is the author's opinion that, in the key process areas described in this paper, TRW-Government IPTs have worked extremely well, have contributed materially to the quality of the BMC3 product, and have validated the IPT concept espoused by this streamlined acquisition effort.

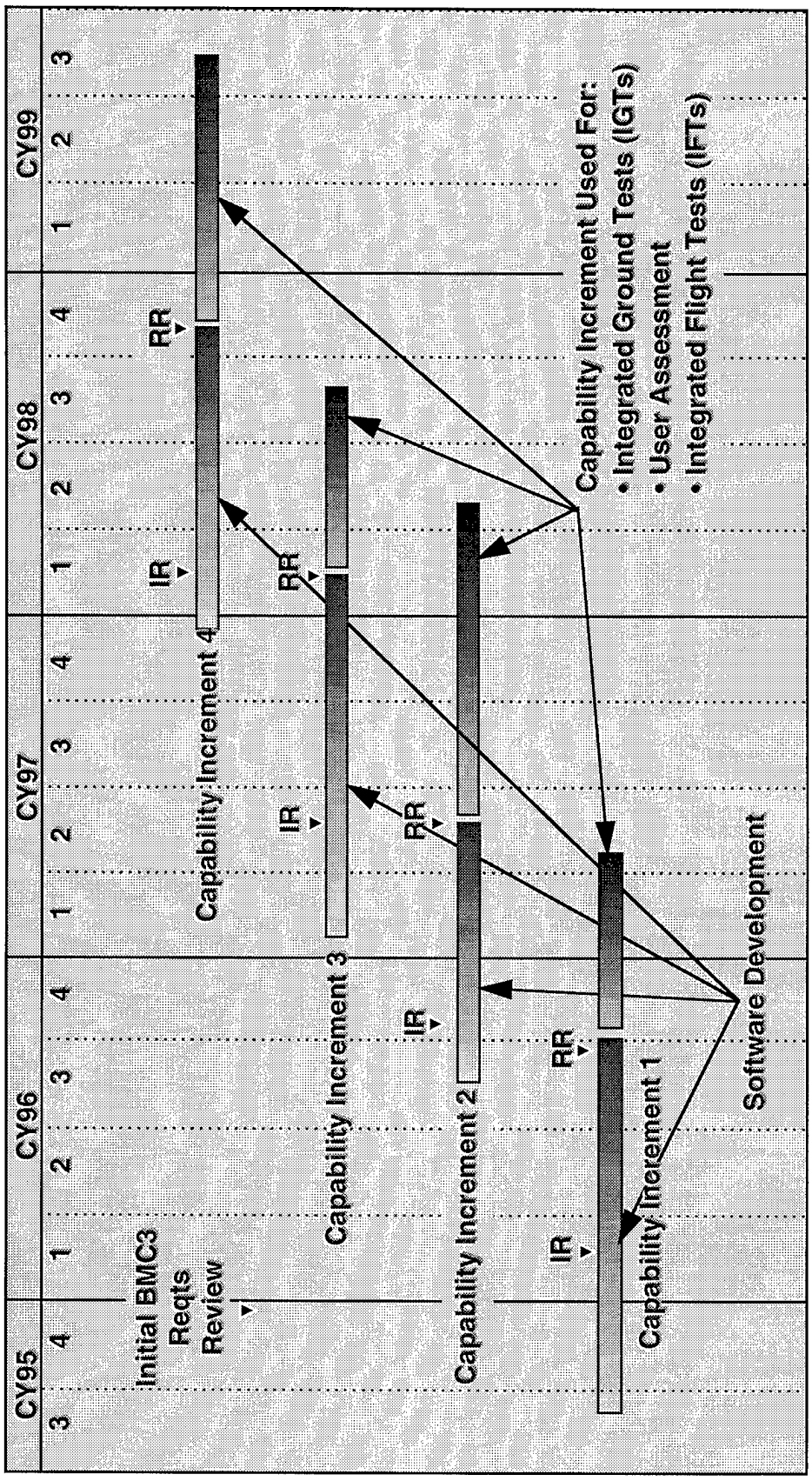

Figure 1: Basic BMC2 Capability Increment Schedule

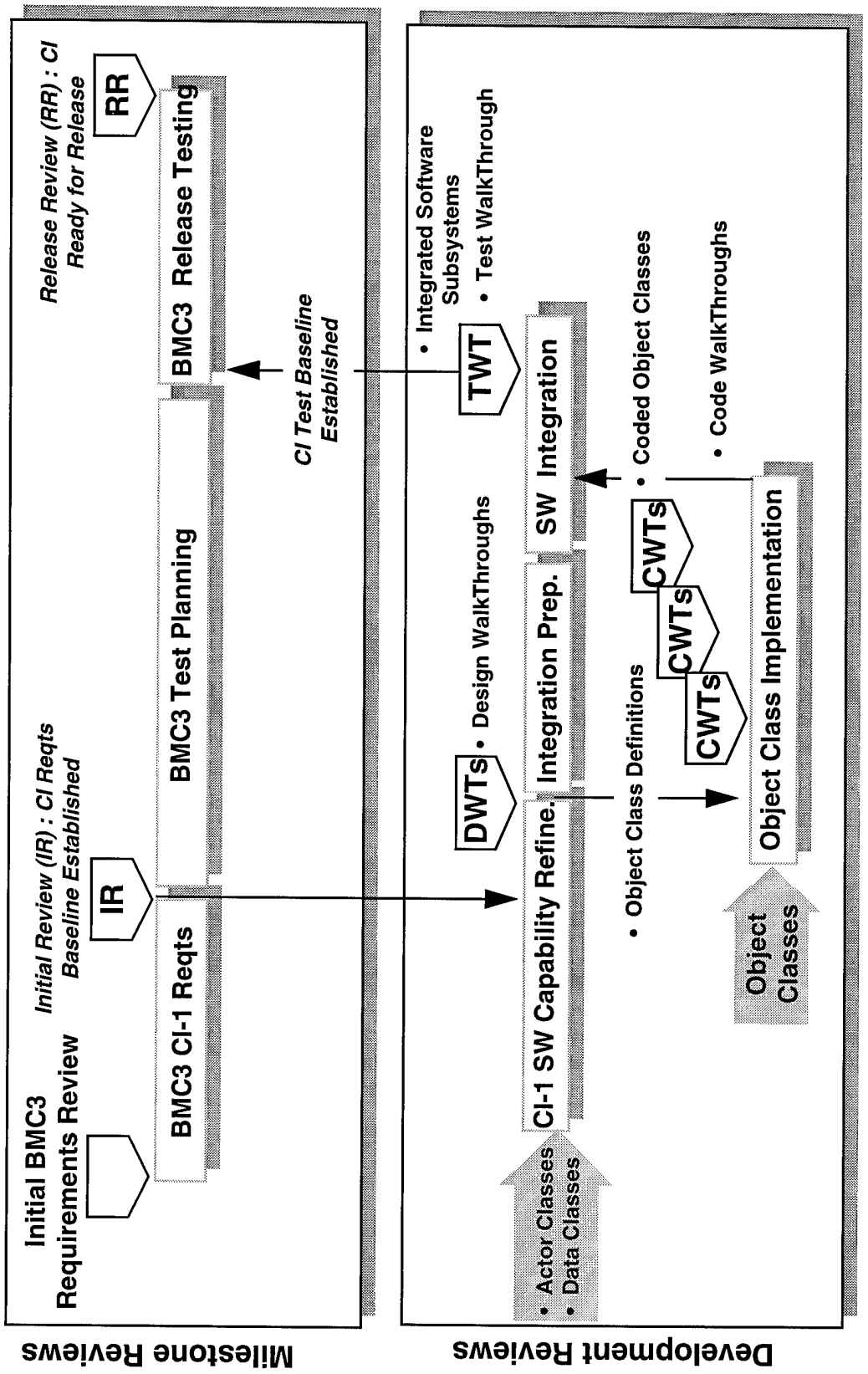

Figure 2: CI-1 Reviews & Walkthroughs

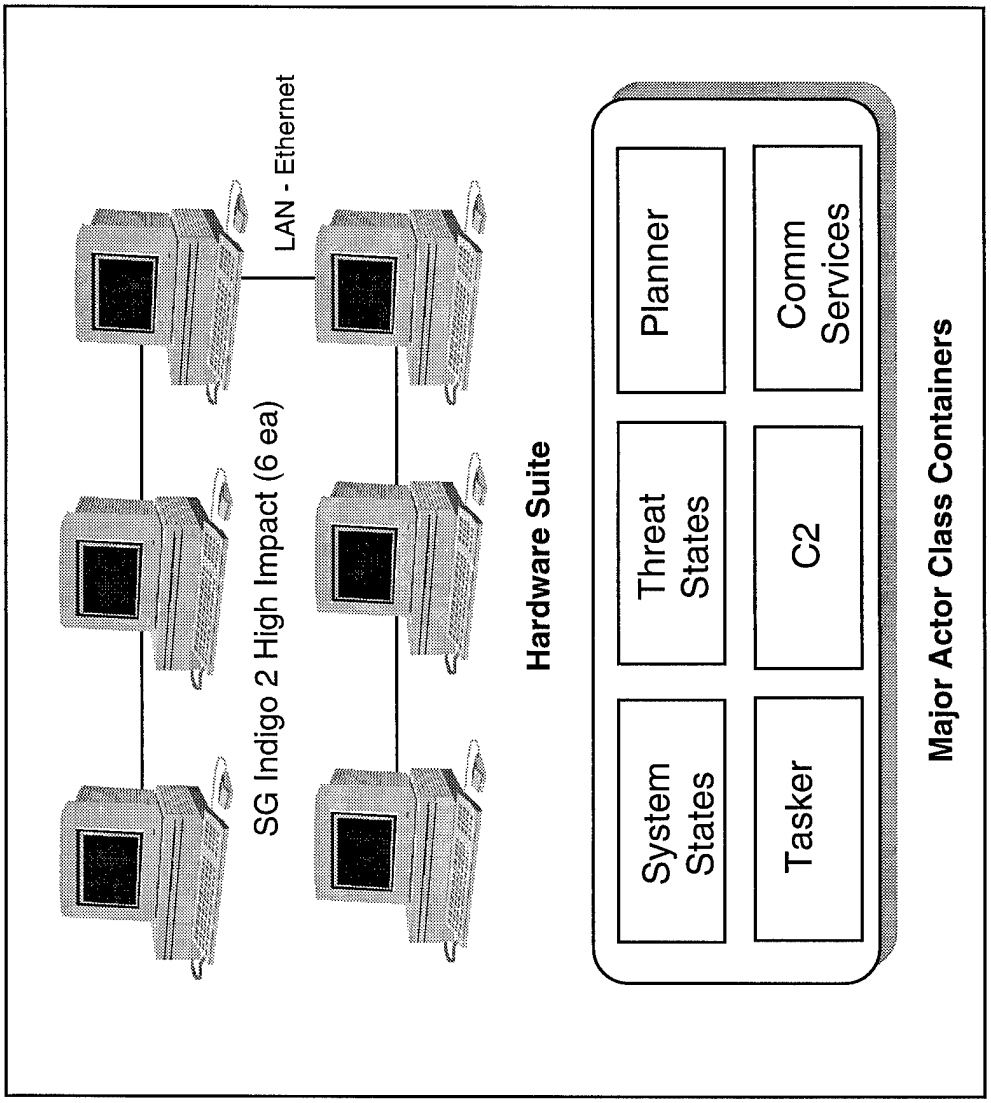

Figure 3 : BMC2

- System States
  - Health & Status of Elements, Including Readiness & Inventory
  - Status of Current Engagement
- Threat States
  - Local and System Track Files
  - Provides GBI Farm - RV Accessibility Data for Planner
- Planner
  - Weapon Target Algorithm & Weapon Tasking
  - Sensor Tasking
  - IFICS Tasking
  - Integrated Resource Management (GBI, GBR, IFICS, SMTS)
- Tasker
  - Message Interface from World Outside BMC2
- Command & Control
  - HIC Interface
  - War Fighting Status
  - Decision Support Software
- Comm Services
  - LAN Monitor & Control

Fig 4 : Major Actor Class Container Functions

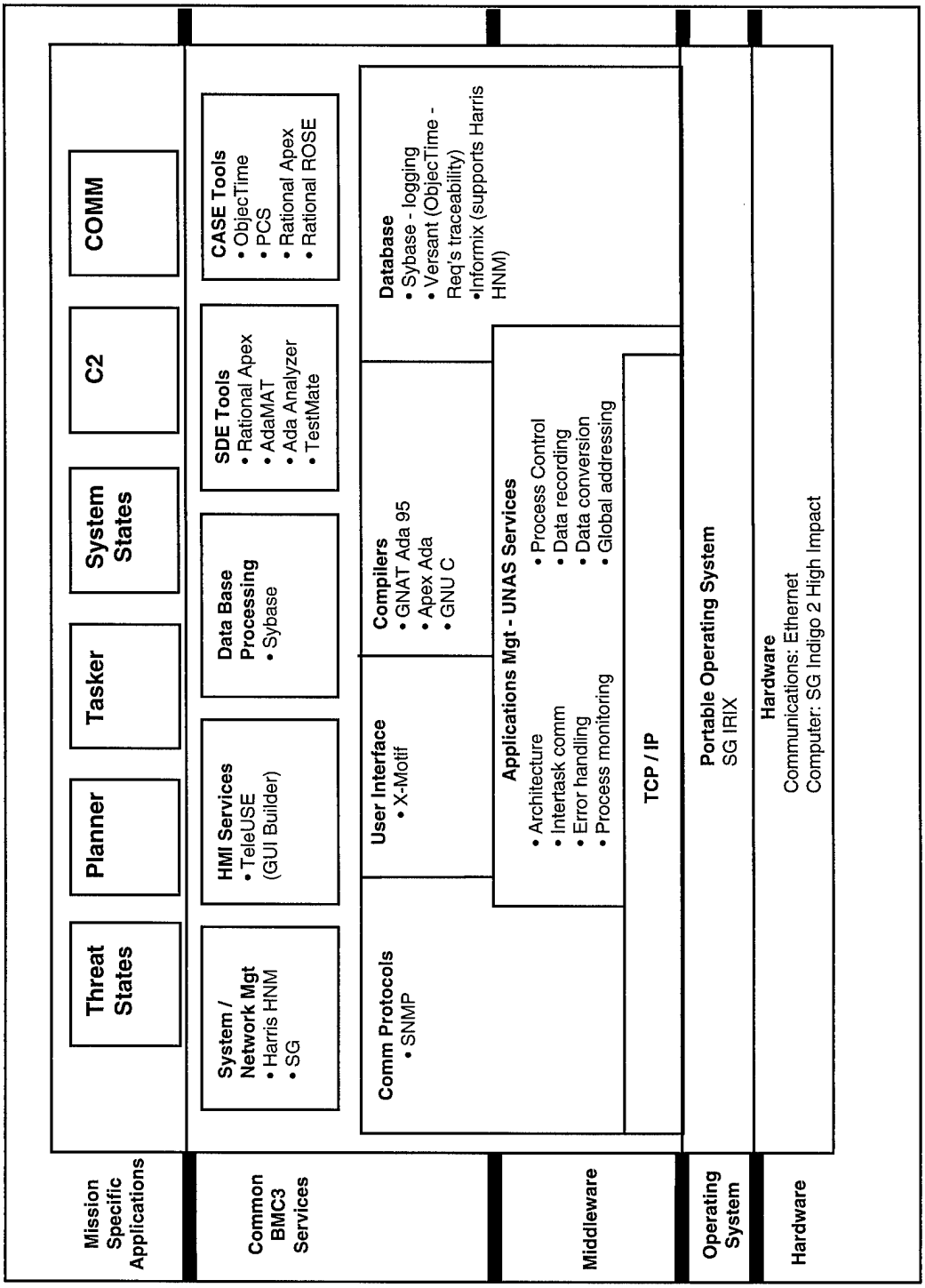

Figure 5: Integrated Engineering Infrastructure

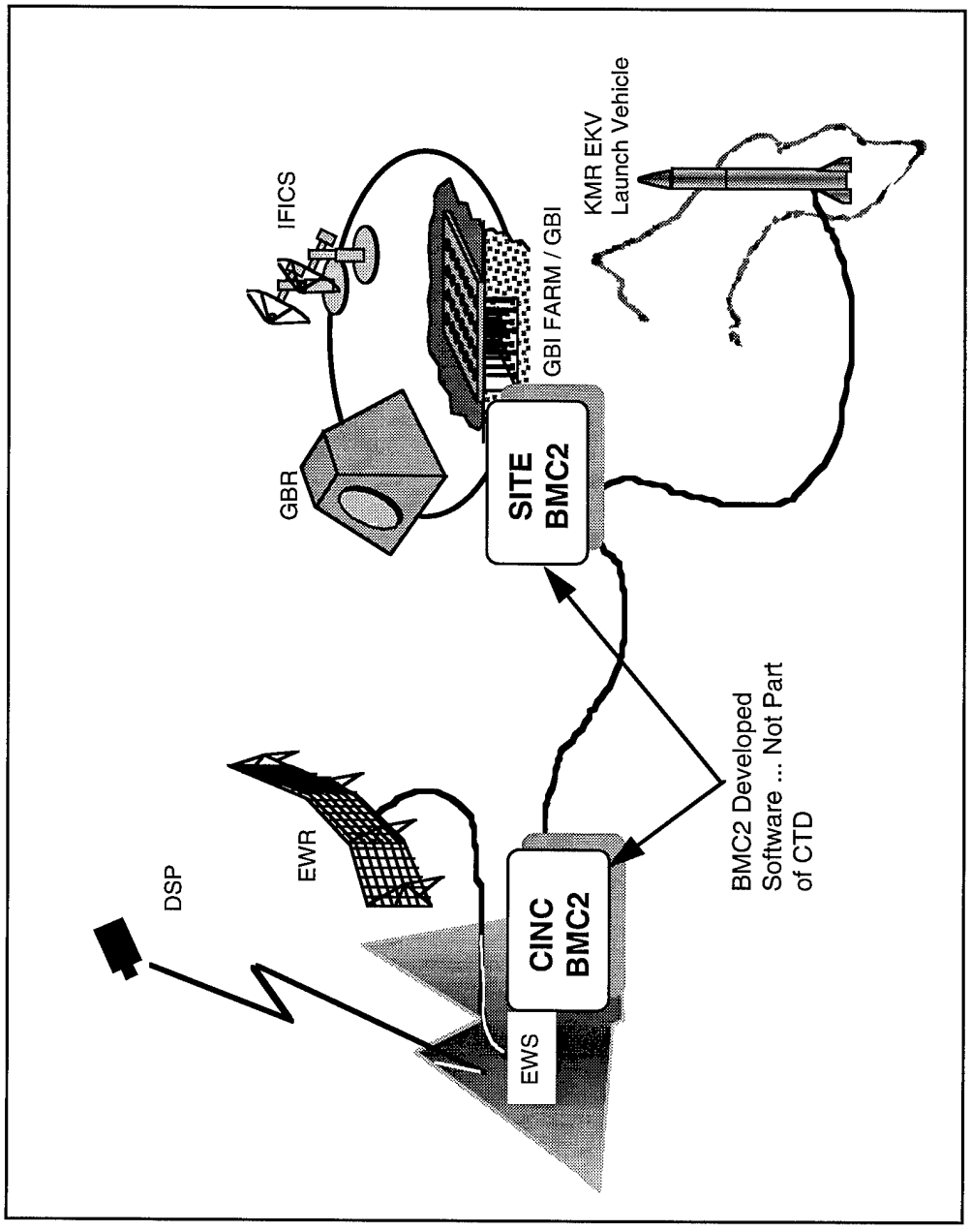

Fig 6: Capability Increment 1, TEx Models Provided as System Drivers

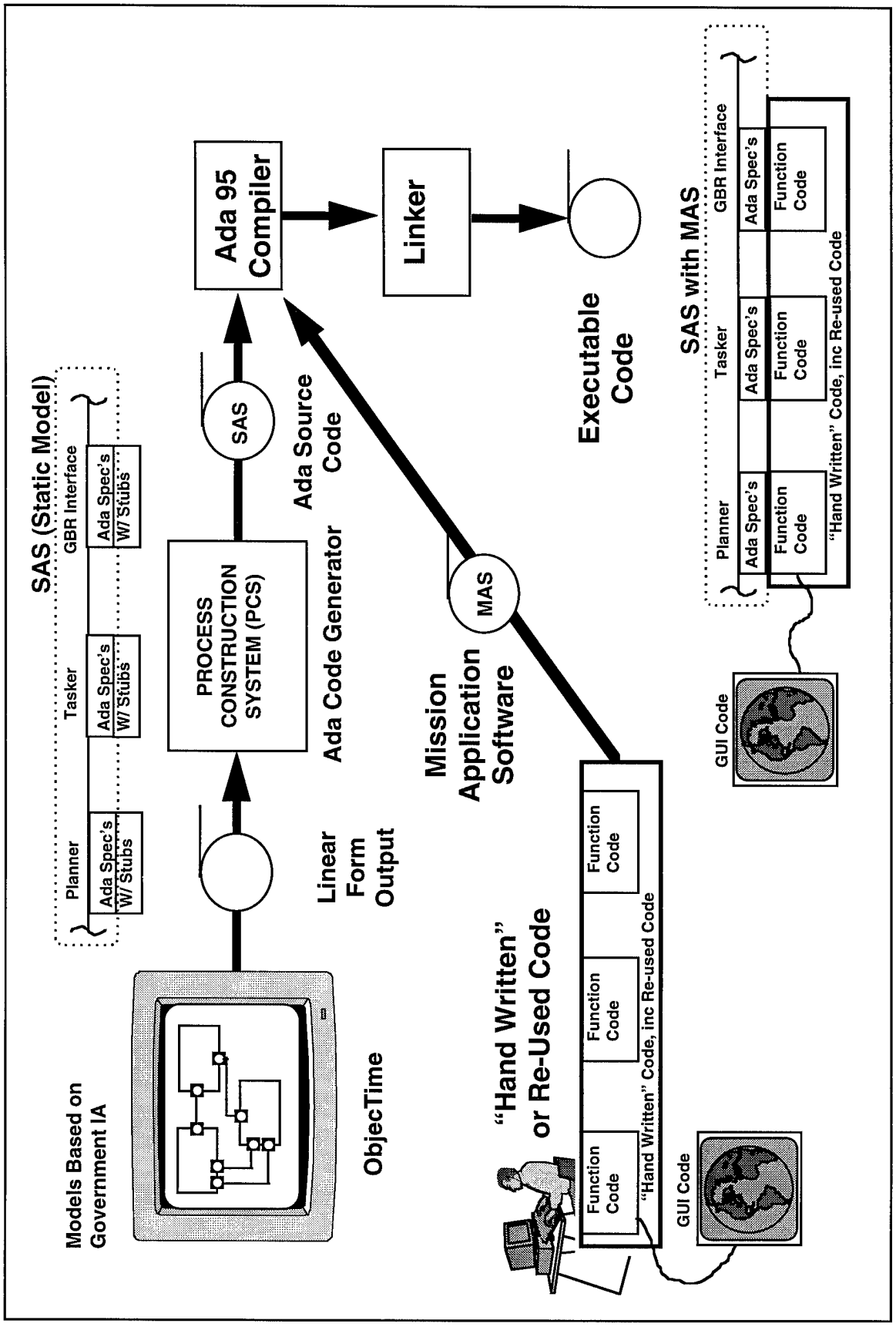

Figure 7 : Software Development Flow

Contractual Milestones: Formal Demonstrations, Release Testing, etc	Tracks major program events and progress	Characteristics: Formal... programmatic & management flavor...large audiences, mostly management & interested parties...extensive prep time...formal vugraph presentations...formally documented (minutes, etc)
Pacing BenchMarks	Shows progress towards Contractual Milestones	Charateristics: Informal... strictly technical...very small audience, engineers only...little or no prep time...no formal vugraph presentations...informally documented (check list only)

Table 1

UNCLASSIFIED

Pass	Sufficient quantity of check list items were passed so as to demonstrate high confidence that the PBM objectives were attained ... check list items that failed were not deemed critical
Conditional Pass	Failed to meet "Pass" criteria, but is deemed readily correctable within cost & at little risk to major program milestones ... only failed items need to be re-run with Government witness / assessment
Conditional Fail	Failed to meet "Pass" criteria, but is deemed correctable with reasonable time & budget relief ... complete PBM must be re-run & passed before proceeding ... management intercession required to help resolve issues
Fail	Clearly failed to show PBM objectives meet ... major cost & schedule impact ... major management intercession required ... this is the train wreck

Table 2

Pass	Start developing Check List for next PBM ... continue on schedule
Conditional Pass	Develop work around plan; schedule failed item(s) re-run. After passing these, start developing Check List for next PBM ... continue on schedule. Advise Government Management.
Conditional Fail	Develop work around plan, re-schedule PBM. After passing these, start developing Check List for next PBM ... continue on schedule. Advise Government Management.
Fail	Stop work, advise Government management, support recovery planning

Table 3

UNCLASSIFIED

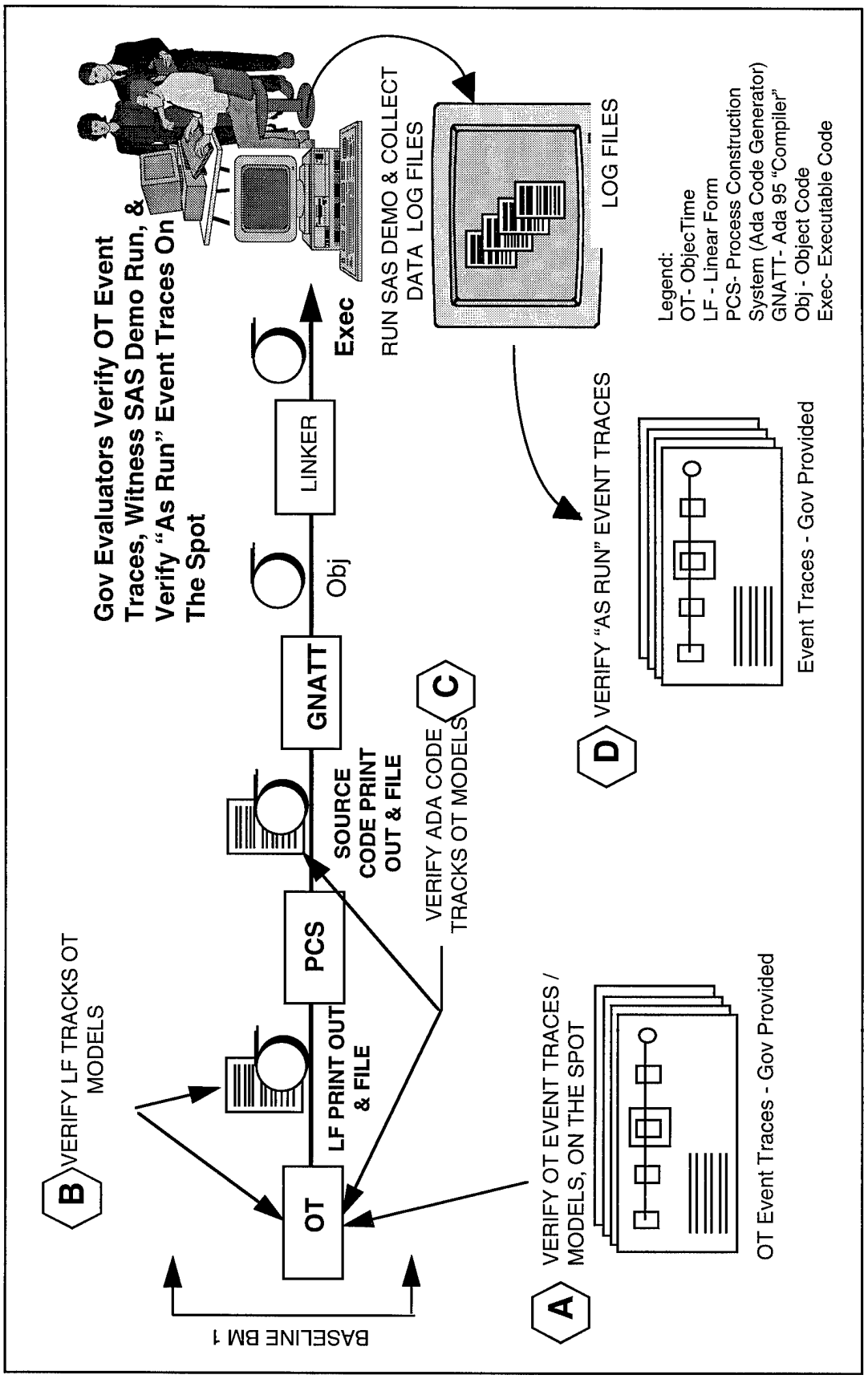

Figure 8: PBM 1 Control Run

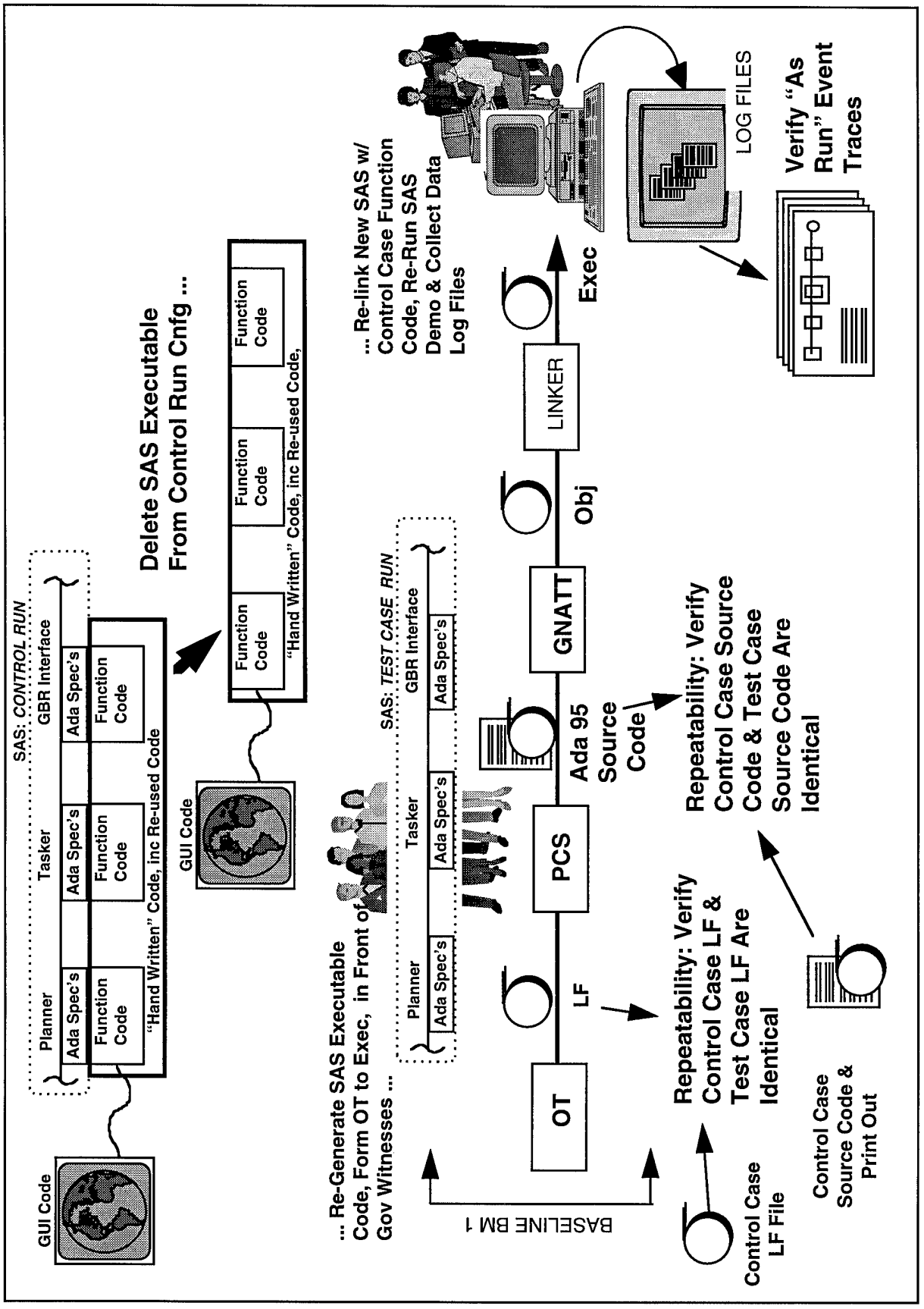

Figure 9: PBM 1 Test Case Run

Total of 319 CI-1 Defects Identified

Severity	Total
- Level 1 (Cause Crash)	5 (2%)
- Level 2 (Cause Major Function To Be Disabled or Incorrect)	15 (5%)
- Level 3 (Cause Minor Function To Be Disabled or Incorrect)	123 (39%)
- Level 4 (Superficial Error)	170 (53%)
- *Level (To Be Determined)*	*6 (2%)*
TOTALS	319

2 Aug 96

Figure 10: Defects Summary

UNCLASSIFIED