ANL--93/1

DE93 007786

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, IL 60439-4801

ANL-93/1

Toward a Methodology for Complexity Management

by

*G. H. Chisholm**

Environmental Assessment and Information Sciences Division

December 1992

*Also of the Mathematics and Computer Science Division, Argonne National Laboratory.

This work was supported in part by the Strategic Defense Initiative Office, Office of the Secretary of Defense, under PMA-2304, and by the National Security Agency under MOD 708992.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

CONTENTS

NOTATION vi
ABSTRACT vii
EXECUTIVE SUMMARY ix
1 INTRODUCTION 1
2 COMPLEXITY MANAGEMENT 3
3 APPROACH 5
4 DEVELOPING A GENERIC BM/C^3 MODEL AND IDENTIFYING CRITICAL FUNCTIONS 7
4.1 Evolution of a Multilevel Model 7
4.2 Validation of the Generic Model 9
4.3 Analyzing the Models 10
4.4 Summary 12
5 THE PROCESS OF DECOMPOSITION AND SELECTION 14
5.1 BLOCKS, LAYERS, CONSTRAINTS, AND ACCEPTANCE TESTS 14
5.1.1 Strategies for Decomposing Critical Functions 14
5.1.1.1 Multiple Blocks 14
5.1.1.2 Multiple Layers 15
5.1.1.3 Multilayered Blocks 15
5.1.2 Additional Constraints 15
5.1.3 Acceptance Testing and Trust Enhancement 17
5.2 A Multilayered Design 18
5.2.1 Firewalls 19
5.2.2 Multilayered Function Allocation 19
5.3 Instantiation of the TCMS with GPALS BM/C^3 Functions 19
5.3.1 Trust 19
5.3.2 Complexity and Functionality 21
5.3.3 Coordination 22
6 SUMMARY AND RECOMMENDATIONS 25

ACKNOWLEDGEMENTS 26
Bibliography 27
Appendix A Validation of a Generic BM/C^3 Model 28
Appendix B AIS Architecture Summary 36

Tables

1	Multilayered Allocations for GPALS BM/C^3 Functions	21
A.2	Three Degradation Strategies	33

Figures

1	A Multilayered Design	6
2	Generic BM/C^3 — Level 1 – Functional Hierarchy	9
3	Generic BM/C^3 — Level 2 – Sense Function	10
4	Generic BM/C^3 — Level 2 – Plan Function	11
5	Generic BM/C^3 — Level 2 – Execute Function	12
6	The Relation between Trust and Complexity	14
7	Strategy I — Multiple Layers	15
8	Strategy II — Multilayered Blocks	16
9	Trust Enhancement	17
10	Conceptual Design — An Acceptance Test	18
11	A Multilayered Design	20
A.1	GPALS Architecture #1 – Level 2 – Sense Function	28
A.2	GPALS Architecture #2 – Level 2 – Sense Function	29
A.3	AEGIS Architecture – Level 2 – Sense Function	29
A.4	GPALS Architecture #1 – Level 2 – Plan Function	30
A.5	GPALS Architecture #2 – Level 2 – Plan Function	30
A.6	AEGIS Architecture – Level 2 – Plan Function	31
A.7	GPALS Architecture #1 – Level 2 – Execute Function	31
A.8	GPALS Architecture #2 – Level 2 – Execute Function	32
A.9	AEGIS Architecture – Level 2 – Execute Function	32
A.10	The SPE Triad — a repeating pattern within the generic model	35

NOTATION

BM	Battle Management
BM/C^3	Battle Management/Command, Control and Communications
CTP	Communications Tasking Plan
GPALS	Global Protection Against Limited Strike
IFTU	In-Flight Target Update
SDIO	Strategic Defense Initiative Office
SPE	Sense, Plan, and Execute
STP	Sensor Tasking Plan
TCMS	Trusted Complexity Management Strategy
TOM	Target Object Map
TSDM	Trusted Software Development Methodology
WTP	Weapon Tasking Plan

TOWARD A METHODOLOGY FOR COMPLEXITY MANAGEMENT

by

G. H. Chisholm

ABSTRACT

This report focuses on the Battle Management/Command, Control, and Communication (BM/C^3) element of the Global Protection Against Limited Strike (GPALS) system. The approach is based on the development and validation of a generic BM/C^3 model. Central to the approach is the tenet that the design is divided into multiple layers. The critical functions make up the bottom layer, where trust is established and significant design effort is required.

EXECUTIVE SUMMARY

This study is concerned with developing a methodology for managing the complexity of the BM/C^3 element of the GPALS system. The premise for our study is that complex systems are prone to failure. Respective to the GPALS system, a failure can manifest itself either as inaction when action is required or as unwarranted action when no action is needed. We are motivated to improve our trust in the system to avoid such failures.

This study concentrates on the functional degradation of the BM/C^3 element of the GPALS system. Specifically, we present one approach for graceful degradation of functionality allocated to the computer resources of the BM/C^3 element. The following are results from analysis of those functions:

- The functions Sense, Plan, and Execute are critical to accomplishing the GPALS mission.
- A function is made up of multiple layers. The inner layer represents a minimal, or degraded, functionality, whereas the outer layer represents the fully evolved functionality.
- The inner layer is made up of those functions that are critical and trusted. Subsequent layers perform less critical, more complex functions with successively lower levels of trust.
- The following design paradigms provide an initial architecture that complies with this approach to complexity management:

  a. No single point of failure shall impair the functionality of the most critical layer (diversity, recovery blocks, and/or partial proofs of correctness combined with testing are recommended);
  b. Firewalls shall ensure that a failure in a less critical layer does not affect a more critical layer; and
  c. Acceptance tests may provide a mechanism for graceful degradation and trust enhancement.

- The enabling computer resources are assumed to be reliable, available, coordinated, and tolerant of single points of failure.

We have identified the following topics for future research and engineering directions:

1. Determination of the critical functions for the trusted computer resources
2. Experimentation to support development of the alternative concepts for firewalls.
3. Demonstration of the multilayered design paradigm. This would be performed in consort with work on the trusted computer resource.
4. Development of a methodical approach for application of a Trusted Complexity Management Strategy.
5. A case study into the application of diverse formal methods to the design of representative, critical pieces of SDIO-GPALS BM/C^3.

1 INTRODUCTION

Increasingly, computers are being used to control life-critical systems. This trend is based on perceived advantages gained from the flexibility assured by programmability. However, the trade-off for this flexibility is complexity and unreliability, and mismanaged complexity often results in unexpected performance — performance that has life-critical implications.

We investigated an approach to complexity management of the Battle Management/Command, Control, and Communication (BM/C^3) element of the Global Protection Against Limited Strike (GPALS) system. We derived a set of requirements from a study of three diverse BM/C^3 architectures: the GPALS architectures developed by the Strategic Defense Initiative Office (SDIO) System Engineering and Integration (SEI) contractor and by the Architecture Integration Study (AIS), and the AEGIS architecture. During this study, we observed that the GPALS architectures concentrated on the allocation of functionality from a systems viewpoint. From this viewpoint, devolution was conceived to affect coordination, for example, failure to provide global coordination of elements or failure in the communication network.

Our specific charter was to analyze the BM/C^3 architecture from a computer resources viewpoint. From this viewpoint, devolution (i.e., graceful degradation with respect to BM/C^3 functionality) must consider those functions allocated to the computer resource (i.e., the underlying computational engine) and those allocated to the software. The functions allocated to the computer resource were less defined than those allocated to the software. Thus, our study concentrated on degradation of the software functionality, where the functionality was more defined. With respect to computer resources, we provide a minimum set of attributes that identify properties assumed during our analysis.

Our approach is based on the development and validation of a generic BM/C^3 model. This model is generic in the sense that it captures functionality that is common to various BM/C^3 architectures. Such a model enables reasoning about the system and determination of critical functions in a generic fashion.

We iterate between reasoning about the evolving model and decomposing the critical functions/subfunctions until reaching a level where the abstraction is sufficiently defined to formulate a specification. This specification provides the basis for managing complexity.

Central to our approach is the tenet that the design is divided into multiple layers. The critical functions make up the bottom, or innermost, layer – one that must be designed with utmost care. It is this layer where trust is estabiished and where significant design effort is required.

2 COMPLEXITY MANAGEMENT

The GPALS system must perform as intended and only as intended, with a high level of trust. Failures have two root causes: (1) design errors and (2) operational faults.

Software-induced failures are attributed to design errors [1], namely, errors in either the design process or in artifacts created during this process. A report by GE Aerospace on Trusted Software Development Methodology (TSDM) [2] suggests that formal methods are useful in the avoidance of errors attributable to the design process. Because of the complexity of the GPALS BM/C^3 system, however, using formal methods to establish a uniform level of trust is virtually impossible (from the viewpoint of both technology and cost).

Avoidance of specification errors is a more insidious problem. That is, the traditional approach to verification and validation (V&V) consists of demonstrating that the artifacts created are in substantial compliance with the specification. Thus, specification errors may not be detected during V&V. A promising alternative is *complexity management*. According to Webster's Ninth New Collegiate Dictionary, *complex* is defined as "a whole made up of complicated or interrelated parts" and *to manage* is defined as "to treat with care."

Complexity management is based on techniques used in the design of life-critical systems. Specifically, these techniques comprise (1) identifying and separating critical functions from less critical functions, (2) implementing these functions in a fashion commensurate with respective trust requirements, and (3) ensuring that less critical functions cannot affect the functioning of more critical functions.

These techniques are predicated on the identification and separation of critical functions from those less critical. The premise is that the critical functions make up a fraction of the total system functionality. This approach to complexity management complements the use of formal methods. That is, by providing separation between critical and non-critical functions, the designers may allocate resources commensurate with trust requirements. A reduction in overall resources, over those required for a uniformally critical system, should result.

Moreover, in recognition of the impact of operational faults, such a separation should comply with the operational constraints imposed by the use of fault-tolerant processing. That is, the current state of the practice in fault-tolerant computing

indicates that a limited functional capability should be allocated to resources that must be tolerant of single points of failure.

3 APPROACH

Any complexity management strategy must depend on fault management. The essence of fault management is (1) recognition of the fallibility of components and (2) the containment of these failures. Fault-tolerant systems design concentrates on containment of operational failures. Here, we concentrate on containment of faults resulting from design errors.

The design of a BM/C^3 element comprises three functions: Sense, Plan, and Execute. All three functions are critical. By *critical*, we mean that if the function is removed from the system, the system will not achieve its objective.

Nevertheless, single instantiations of a Battle Manager (BM) function may not be critical. This apparent paradox results from the fact that redundancy permeates the GPALS system through replication and degradation strategies (both local and global). We can conceptualize this redundancy as a series of concentric layers, shown in Figure 1. Each layer corresponds to a level of functionality. The outermost layer represents the fully evolved functionality; the innermost layer represents minimal, or devolved, functionality.

The advantage of this multilayered approach is that it reduces the difficulty of assuring trust for a system. Only the innermost layer is made up of the functions that are critical and trusted. The outer layers perform less critical, more complex functions with lessening trust requirements. All the layers are separated by so-called firewalls (control and data) to ensure that failures in less critical layers do not obviate the functionality of a more critical layer.

In developing a complexity management scheme for the software, we have made one essential assumption about the enabling computer resources (i.e., hardware, operating system software, and network):

The computer resources are designed and implemented commensurate with the BM/C^3 trust objectives: that is, they are reliable, available, coordinated, and tolerant of single points of failure.

With this essential assumption, we now present our approach to complexity management for the GPALS system. For the sake of completeness, we include a discussion of the evolution of a generalized BM/C^3 model (based on a study of three diverse BM/C^3 architectures). We then show how we validated this model and identified critical functions of the GPALS BM/C^3 system from this model.

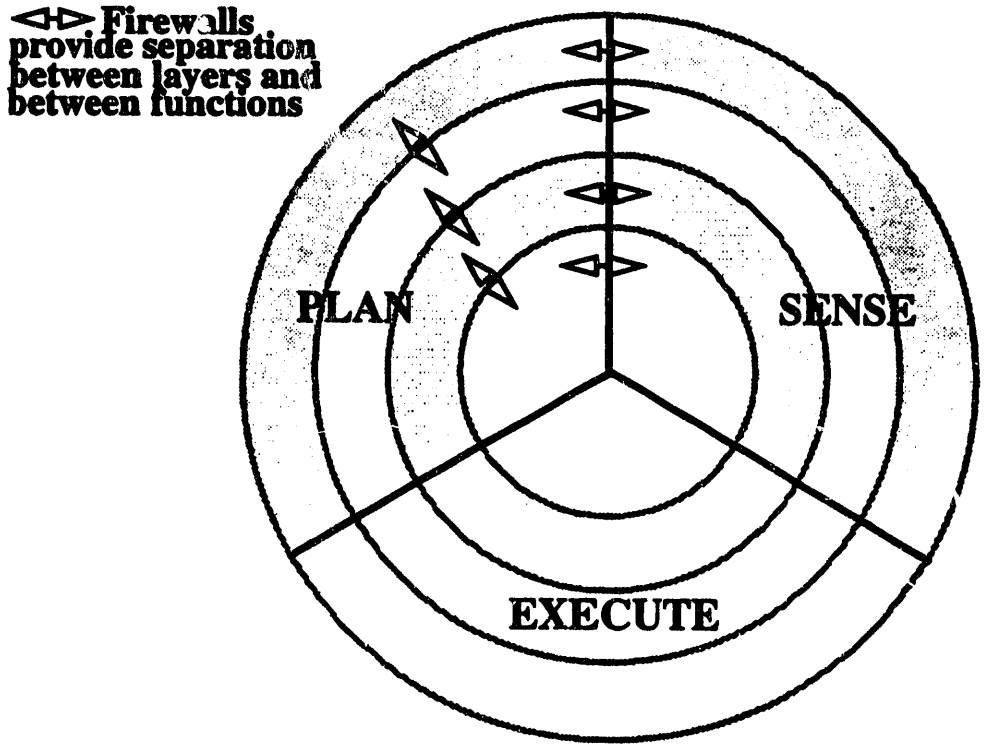


Figure 1 A Multilayered Design

4 DEVELOPING A GENERIC BM/C^3 MODEL AND IDENTIFYING CRITICAL FUNCTIONS

This section describes a generic BM/C^3 system. A *generic model* is one from which multiple implementations may be instantiated. In this sense it captures the functionality of the system at a conceptual level. We define a properly conceived model as one that provides the basis for the highest possible level of functionality, is free of design decisions, and thus forms the basis from which multiple implementations may be derived.

Additionally, a generic model enables individuals (e.g., engineers, scientists, and managers) to reason about a system being designed, identifying essential properties of the system, and comparing trade-offs involved in generating a system specification.

A system specification is derived from such a model. This derivation starts with analysis of the model. From this analysis a number of alternative decompositions are proposed. The appropriate decomposition is selected from the alternatives by careful consideration of their respective functionality and compliance with design constraints. These steps are repeated until all functions are a least reducible unit (see Appendix A).

The remainder of this section describes the development of a generic model for the GPALS BM/C3. The following section presents one pass through the description of the remainder of the process of developing a generic model.

4.1 Evolution of a Multilevel Model

We begin by forming a function diagram of the generic system. Figure 2 depicts the generic system at this level. Three main functions are identified: Sense, Plan, and Execute. The properties and functionality associated with each function are as follows:

1. Sense

Figure 2 models a fully coordinated GPALS system that is made up of multiple sensors. A fully devolved system, or enclave, is also modeled by a single sensor. Further, the devolution of the dispersed system is modeled as follows:

a. Properties

- Detect and transmit only threats, and

- No spurious threats.

b. Functionality

- Inputs — Sensor Tasking Plan (STP) and signal energy.
- Outputs — Vectors.
- Function — signal energy is associated with objects in the environment. This energy is converted into an internal representation depicting threats.

Figure 2 also depicts the functionality associated with the Plan and Execute functions.

2. Plan

Plan is the heartbeat of the GPALS BM/C^3. Note that we have modeled one planner. This depicts the need for planning whether the system is operating in a fully coordinated mode, in enclave mode, or in a mode between these extremes, as shown below:

a. Properties

- Provide a correct plan, and
- No spurious plans

b. Functionality

- Inputs — CINC commands, vectors, system health and status.
- Outputs — STPs, Weapon Tasking Plans (WTP), and CINC information
- Function — given health/status, vectors, and CINC commands, develop STPs and WTPs that best meet system objectives.

3. Execute

Plan accomplishes the coordination between execution elements (weapons), and Execute models the functionality associated with executing the WTPs.

a. Properties

- Execute on command, and
- No spurious execution.

b. Functionality

- Inputs — WTP.
- Outputs — Weapons and health/status.
- Function — execute the WTP to best meet the system objectives.

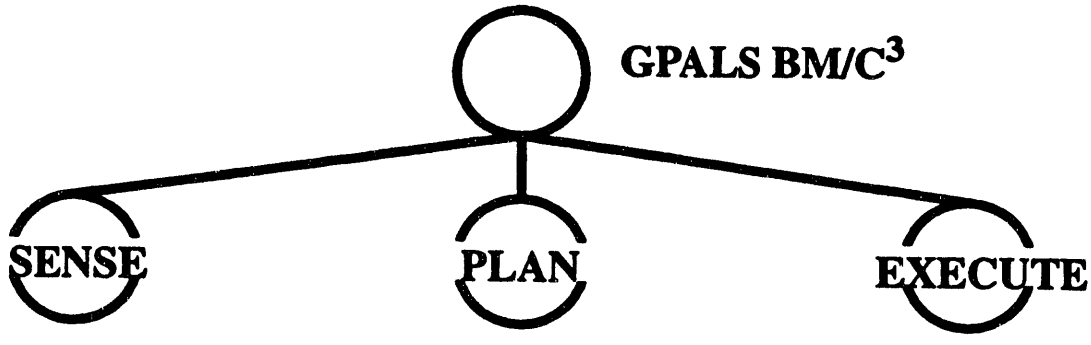


Figure 2 Generic BM/C³ — Level 1 - Functional Hierarchy

Each function can then be subdivided further. Figure 3 depicts the Sense function of the Level 2 functional hierarchy. Similar representations are developed for the Plan function and the Execute function in Figures 4 and 5, respectively.

4.2 Validation of the Generic Model

The next stage in our study involves validation. Appendix A depicts the functional decomposition of the three BM/C³ architectures studied. The close correlation between our generic model and the BM/C³ architectures provides an informal validation of the model.

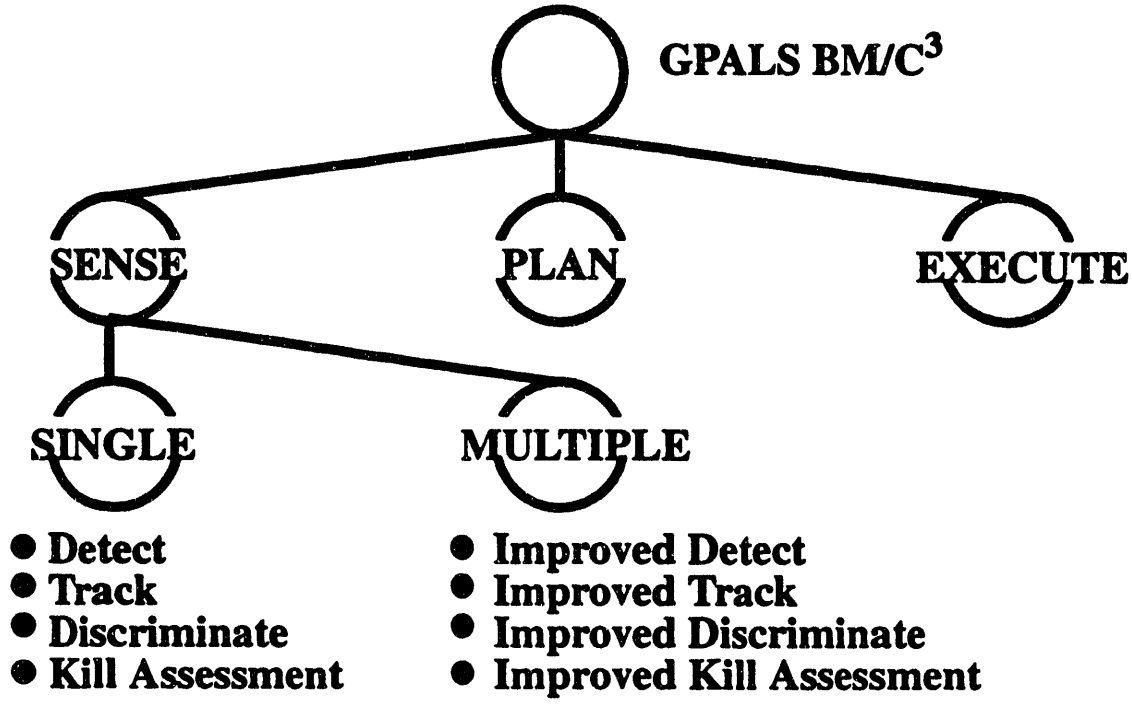


Figure 3 Generic BM/C³ — Level 2 - Sense Function

4.3 Analyzing the Models

The final stage in our study involved analysis of the model and identification of critical functions. To this end, we compared two different GPALS architectures.

1. In the first architecture, devolution is based on the assumption that communications are assured within individual sites, but that a site may either become isolated as a result of communications degradation or loss) or nonfunctional. There are two basic modes of operation: fully coordinated and degraded, with the degraded mode having varying degrees of functionality.
2. In the second architecture, devolution is based on decreasing functionality for communication. Data is flood-routed (pruning accommodates degradation).

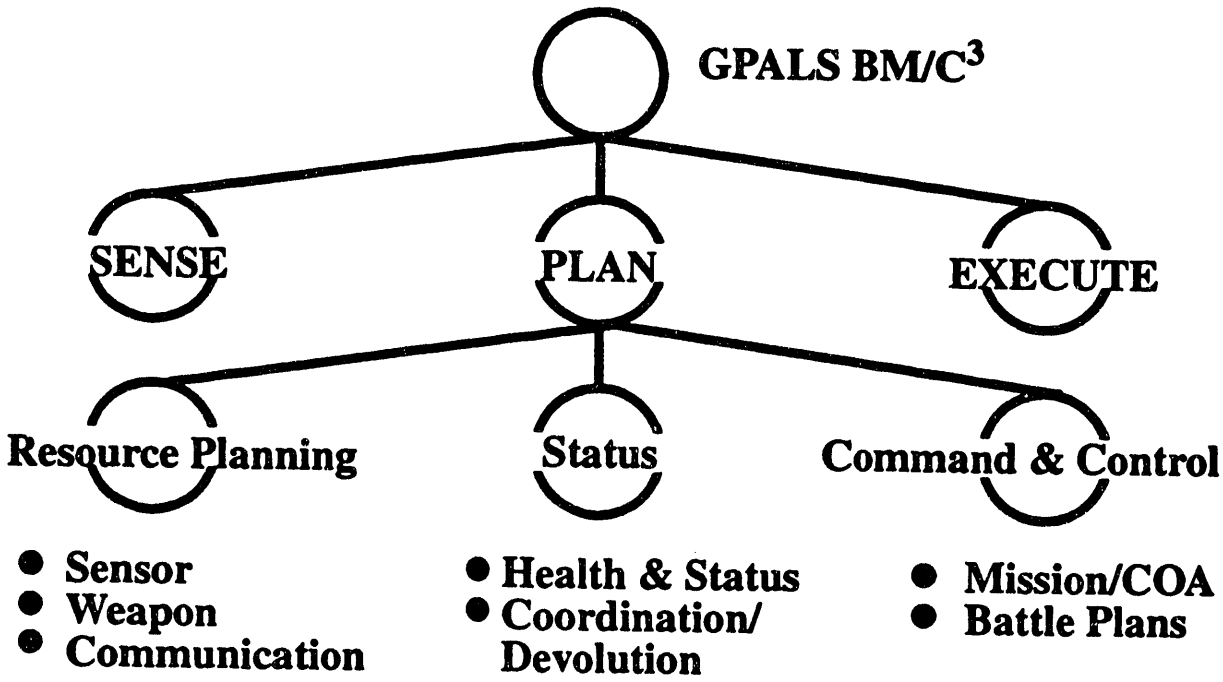


Figure 4 Generic BM/C³ — Level 2 – Plan Function

Moreover, there are three modes of operation: fully coordinated, partially coordinated, and autonomous.

3. Both the architectures address *global devolution.*

During this phase of the case study, the AEGIS architecture was omitted. We chose this approach because of architectural differences that are predicated by substantially different missions. Appendix A provides additional detail respective to reasoning about the BM/C³ models.

As a minimum, devolution strategies involve accommodation of two design constraints. The first is that no single point of failure shall obviate the success of the mission. The second is that the system shall anticipate the occurrence

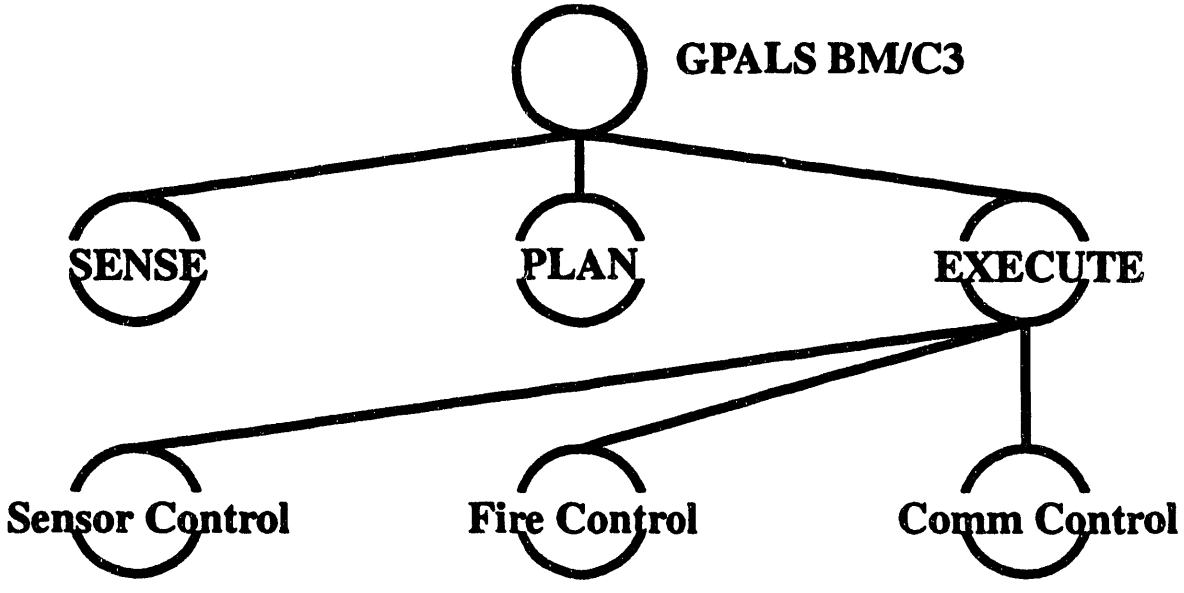


Figure 5 Generic BM/C^3 — Level 2 - Execute Function

of operational failures and gracefully make a transition from a fully operational capability to a lesser capability. These constraints effect the design decisions embodied in the next level of decomposition.

The next section describe these effects in the context of the further decomposition of the SPE triad. This decomposition is pursued as the respective complexity of the SPE functions (Level 1) exceeds that associated with components of comparable reliability and availability required for the GPALS BM/C^3.

4.4 Summary

At this juncture in our study, we have identified the critical functions as the

Sense, Plan, and Execute (SPE) triad. However, the complexity associated with these functions is judged excessive respective to known methods for attaining the GPALS trust and operating goals, specifically performance for fault-tolerant computers and use of formal methods for software development.

Therefore, further decomposition is warranted. In anticipation of this decomposition, we have identified properties of the subsequent architecture. These properties are (1) no single point of failure and (2) graceful degradation. In addition, we have differentiated between the application for the GPALS BM/C^3 and the support or enabling subsystems. These subsystems must also possess the two cited properties. This requirement implies that all processing must be free of single points of failure. Such technology is found in the literature on fault tolerance.

Our strategy must be compatible with command devolution strategies and must support graceful degradation within a single BM that is free of single points of failure. An artifact of this support may provide a mechanism to facilitate global devolution and coordination. Consider that each BM has autonomy over its degradation. Coordination between redundant BMs requires a moderately simple voting mechanism that is tolerant of faults and capable of selecting the most capable of the equally trusted BMs. Note that this approach removes a dependence upon the communications system. In contrast, note the potential dichotomy in a system that devolves on the loss of communication and that relies on the communication system to provide a service that harbors no single point of failure.

The following section describes one strategy/architecture for further decomposition of the SPE functionality that is compatible with these properties.

5 THE PROCESS OF DECOMPOSITION AND SELECTION

5.1 BLOCKS, LAYERS, CONSTRAINTS, AND ACCEPTANCE TESTS

Recall that graceful degradation and command devolution are predicated on developing an architecture or strategy that accommodates failure scenarios. Implicit in such an accommodation is the premise that there is an indirect correlation between complexity and trust (depicted in Figure 6).

Reasonable constraints on the acceptability for a particular strategy to adequately manage complexity are (1) the resulting design must be less complex than an unmanaged approach, and (2) trust is improved over the unmanaged approach.

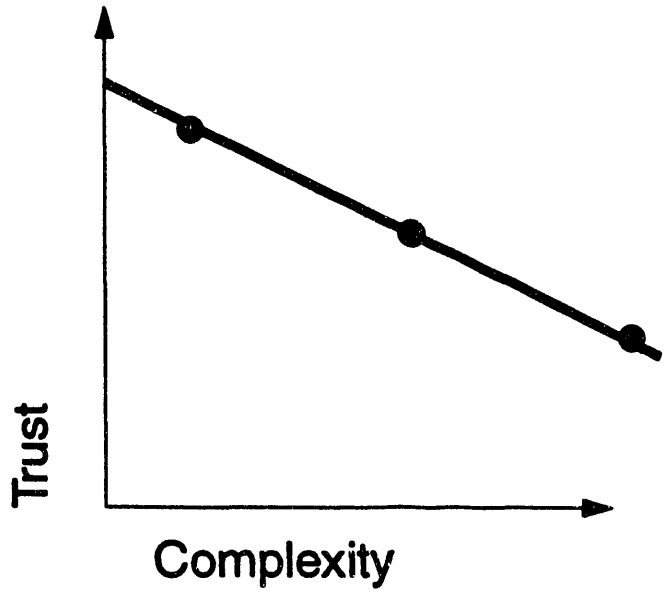


Figure 6 The Relation between Trust and Complexity

5.1.1 Strategies for Decomposing Critical Functions

The following describes alternative approaches to managing complexity.

5.1.1.1 Multiple Blocks One approach would be to divide the system in a number of blocks, the complexity of each would be within manageable limits.

5.1.1.2 Multiple Layers Figure 7 depicts a multiple layered strategy for decomposition. Implicit in this approach is the assumption that the functionality may be divided into independent layers. This figure depicts the Sense functionality as being decomposed into four subfunctions.

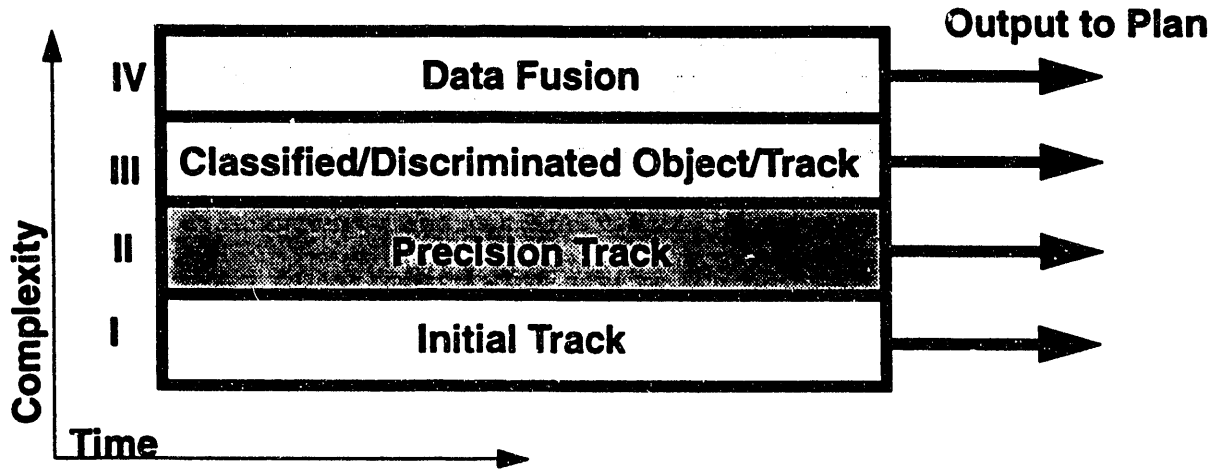


Figure 7 Strategy I — Multiple Layers

5.1.1.3 Multilayered Blocks Figure 8 depicts a multilayered and blocked strategy for decomposition. This is distinguished from the previous strategy by a decomposition of the Sense functionality that identifies a relationship between the subfunctions, yet identifies independence between the constituents of those subfunctions.

5.1.2 Additional Constraints

Both the multilayered and multilayered block alternatives may provide adequate complexity management, but both impose further design constraints. Specifically, a multilayered design that is based on accommodation of design-specific failure scenarios must incorporate a capability for failure detection within a layer and for isolation of failure effects between layers.

The goal for a layered approach in depicted in Figure 9. Our goal is to improve the trust in the system without a reduction in functionality (recall that complexity

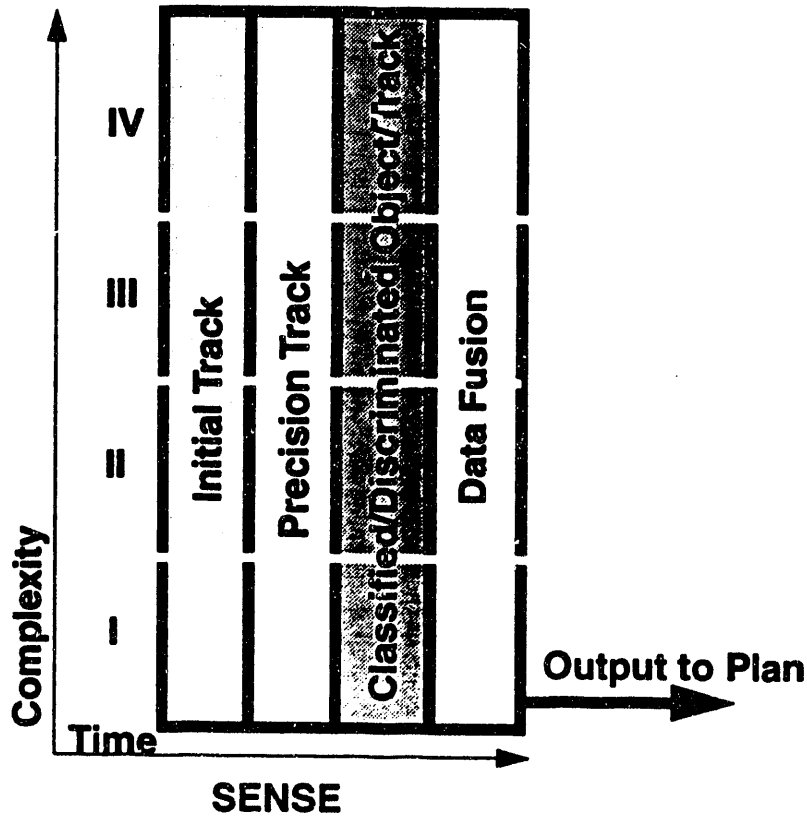


Figure 8 Strategy II — Multilayered Blocks

is proportional to improved fidelity associated with an individual function). We propose a methodology for trust enhancement that is based on *firewalls*.

The minimum capability associated with a firewall implementation is the assurance that the control and data flows of a more trusted layer are not impaired by those of a less trusted layer. This is accomplished by the specification of a layered data structure, the absence of global data, and a verifiable control schema.

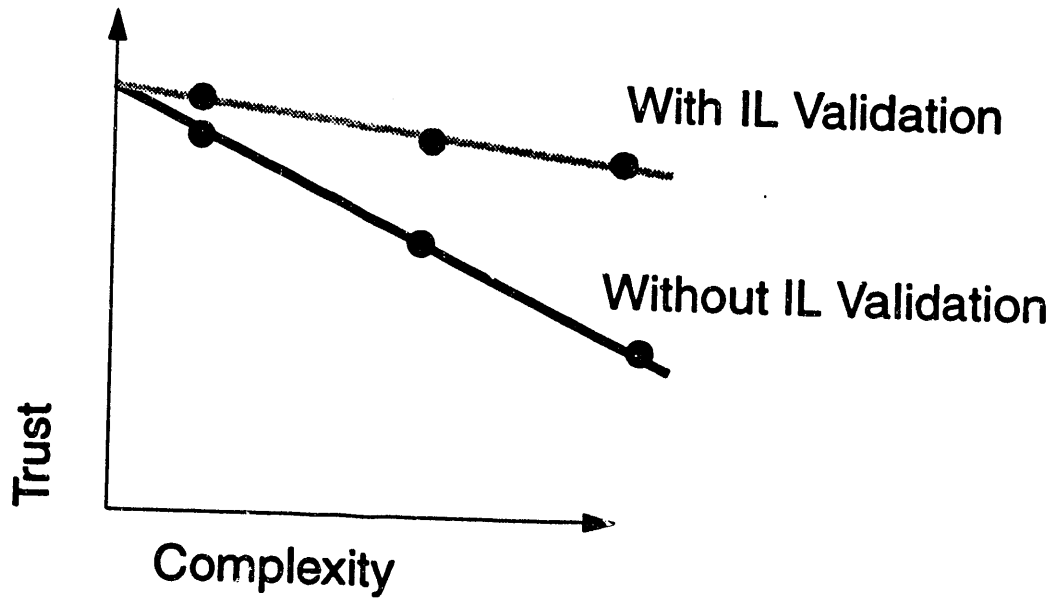


Figure 9 Trust Enhancement

5.1.3 Acceptance Testing and Trust Enhancement

The intuition behind the firewall approach is that a less trusted layer cannot obviate the functionality of a more trusted layer. One method for provision of this capability is to assure that data generated by a less trusted layer does not affect results in the more trusted layers.

With respect to provision for the support of local degradation, the firewall must provide a capability for testing the results of less trusted layers. Figure 10 depicts such a test.

In Figure 10, layers 1 and 2 represent the outputs from a trusted layer and from an adjoining, less-trusted layer. Conceptually, the acceptance test is made up of the following three elements:

1. level consistency,
2. internal consistency, and
3. acceptance.

Level consistency is concerned with a determination that the data generated by the less trusted, more complex function is at least as good as that associated with the more trusted layer. *Internal consistency* is concerned with a determination

that the less trusted results are consistent with the input data. Acceptance may be viewed in terms of intended usage (i.e., multiple blocks or layers). In the following example, multiple layers will be selected. For this example, the output from a function, Sense, will be the result obtained from applying the acceptance test to all layers. In this way, the result will be at a level of trust commensurate with the most trusted layer and the fidelity of the output will be that associated with the most functional layer that passes the test.

It is incumbent on the design to assure that the conclusion drawn by the acceptance test generates a result that is at least as good as that of the most trusted layer.

Note: a correct implementation of independence is dependent on the trusted computer resource (see Section 3).

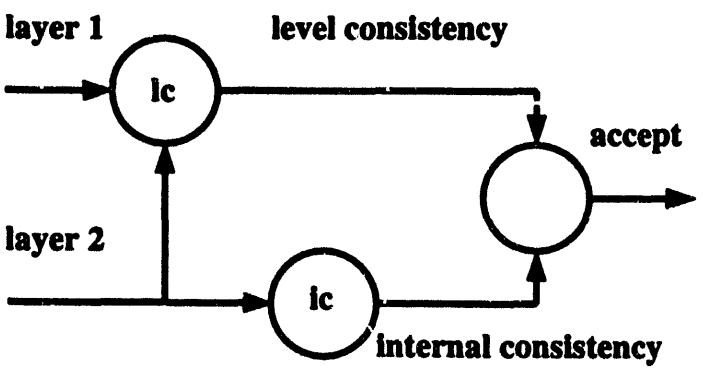


Figure 10 Conceptual Design — An Acceptance Test

The following example provides a case study of the implementation of a multilayered alternative to decomposition as applied to the GPALS BM/C^3.

5.2 A Multilayered Design

Figure 11 depicts a design paradigm that may be used to develop a BM/C^3 specification to meet the requirements identified during our study. Specifically, such a specification must address trust, graceful degradation, and global coordination/devolution. The following describes this paradigm.

5.2.1 Firewalls

An essential feature of this model is that each layer is separated by a *firewall*. These walls ensure that failure of one layer will not impede functioning of another, more critical layer. Because of their importance, the firewalls must be implemented at the highest trust level. Independence of control and data is assured by formal specification, formal verification, and testing.

5.2.2 Multilayered Function Allocation

A second essential feature of this model is definition of the functionality associated with the multiple layers for the Sense-Plan-Execute (SPE) model. Table 1 identifies typical allocations of multilayered functions in the context of a GPALS BM/C^3. This table considers a multilayered decomposition of the GPALS BM/C^3 functionality.

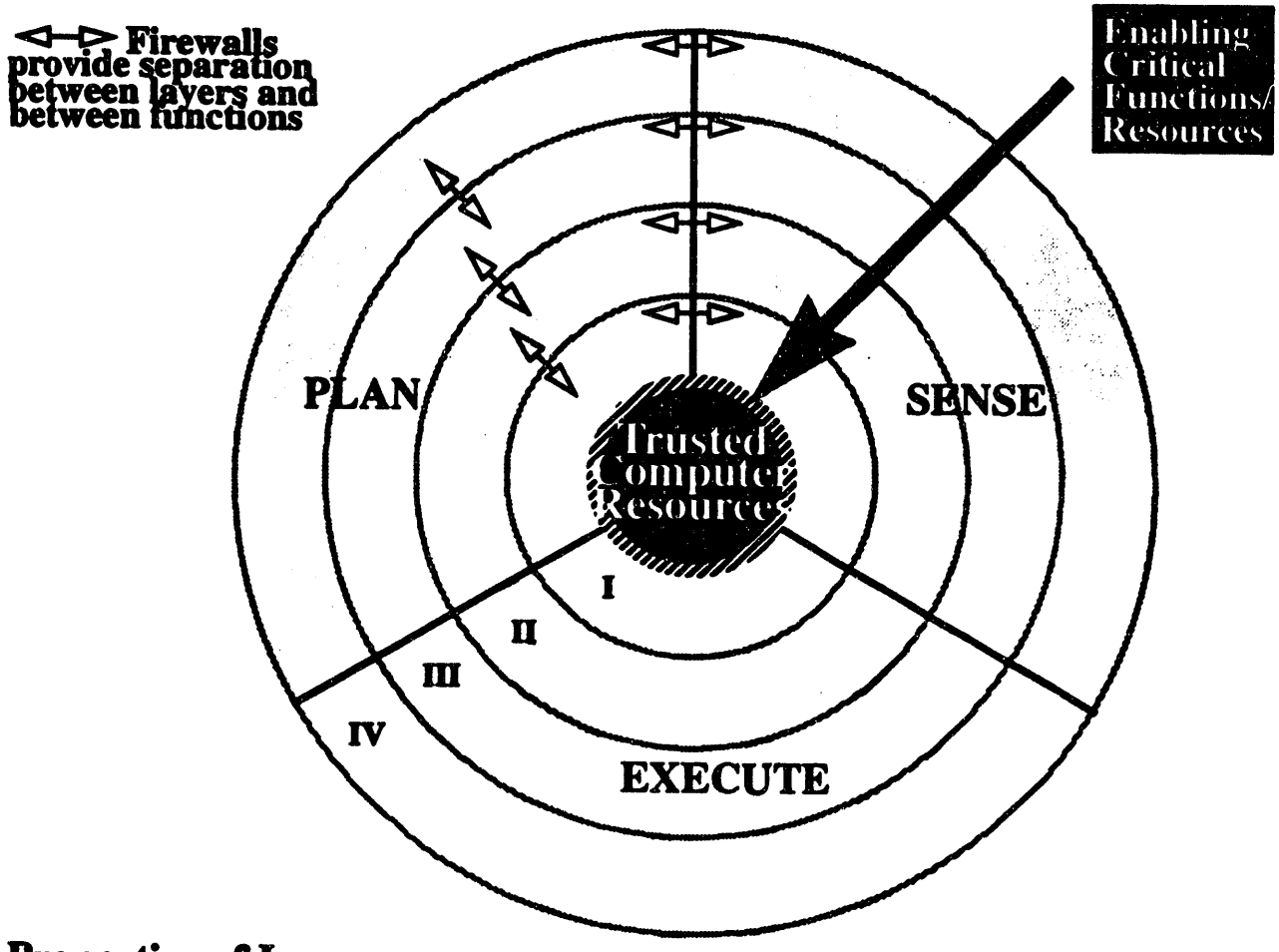
5.3 Instantiation of the TCMS with GPALS BM/C^3 Functions

The intent of this section is to instantiate GPALS BM/C^3 functions within the TCMS shown in Figure 11. These instantiations are provided as a frame of reference for the specification and design of such a system.

Four layers are depicted in Figure 11 and Table 1. The criterion used for allocating functions is based on perceived complexity. The most trusted layer is made up of the least complex functionality associated with a particular function. In recognition that this functionality produces less desirable results, the firewall paradigm incorporates a methodology for acceptance of less trusted results. The following section describes this methodology.

5.3.1 Trust

Table 1 details one decomposition of the GPALS BM/C^3 functions as described by the generic model. The arrow on the left margin shows the direction for graceful degradation of these functions. Layer I is made up of the most trusted functions: those that are the subject of detailed design verification and validation (V&V). This V&V may rely on the use of formal methods to aid in their development and accreditation. Layer IV is made up of the least trusted functions. It is intended that firewalls provide a mechanism for enhancing the trust in the results from the less trusted layers.



Properties of Layers
Outter layer provides higher fidelity with less trust
Inner layer provides lower fidelity with greater trust
Lower layer provides acceptance criteria for outer layer

Figure 11 A Multilayered Design

Table 1 Multilayered Allocations for GPALS BM/C^3 Functions

Trust	Sense	Plan	Execute
Layer I - Most Trusted; Minimal Functionality; Fully Degraded	1. Detect 2. Initiate Track 3. Report	1. Access	1. Launch 2. Fly 3. Home
	4. Track Update	2. Improved Access	
	5. Discriminate 6. Kill assessment	3. Further Improved Access	4. Receive IFTU/TOM (STP, WTP, CTP - implies feedback to sense)
Layer IV - Least Trusted; Maximum Functionality; Fully Coordinated (No Degradation)	7. Data Fusion	4. Global Coordination	5. Global Coordination

DEGRADATION

5.3.2 Complexity and Functionality

Intuitively, there is a indirect correlation between trust and complexity, namely, that complexity is the antithesis of reliability and trust. In our estimation,

there is a direct correlation between complexity and functionality: higher-fidelity computations and/or system actions are associated with greater system complexity. Establishment of trust is dependent on minimization of complexity and, thus, functionality. Layer I functions are less complex than the outer layers. Correspondingly, Layer I functions have minimal functionality.

5.3.3 Coordination

The layering of the sense function is depicted in Table 1. This amplifies the decomposition outlined in Table 1.[1]

1. Layer I (Most Trusted) —

   a. Detect a target
   b. Initiate a track for this target
   c. Provide track information to the planning function

   The intuition is that this comprises the minimum acceptable functionality for the sense function and, therefore, minimizes the effort associated with verification and validation. However, this minimal functionality may provide tracks with a large error envelope. The intent of the inner layer is to provide a trusted result. This result provides a minimum capability to fight a battle and also a standard by which to evaluate the results from an outer layer.

   The functions allocated to subsequent layers provide refinement to the initial track information, and, upon acceptance, enhance the Sense function. However, in the event of BM/C3 degradation, the innermost layer must function. The functionality associated with this conceptual function is

   a. No false alarms, and
   b. No missed detections.

2. Layer II —

   a. Update the track of the target

      The inner layer is based on firing weapons with only the initial detection information. The error associated with the initial detection may be reduced by acquiring subsequent detection(s) and updating the track.

[1] This listing is provided as an example implementation of the Sense function. The intent is to validate the concept, in contrast to presenting an engineered design approach. During development of these concepts, each function was treated in a collective manner. That is, the Sense function may represent one or a collection of sensors and their respective processing.

Acceptance of this output must assure that the updated track is at least as good as the initial track. This implies that no false alarms nor missed detections result.

3. Layer III —

A significant change in system level complexity is noted between Layers II and III. The functions at this layer require activation of system components to accomplish their goal. Specifically, the discrimination and kill assessment functions require sensor tasking to acquire the data to complete their calculations. This affects the planning and execute functions of the BM as well as the control and feedback from the elements.

a. Discriminate between targets and decoys

This function is made up of the subfunctions required to adequately provide discrimination among targets, decoys, and other background effects.

b. Kill assessment

This provides the capability of assessing which targets have been destroyed provides input to a refinement of the battle plan. This refinement helps conserve resources.

4. Layer IV —

Data fusion

The ultimate refinement in the sense function enables the fusion of data from all available sensors and a more definitive track determination.

The layering of the plan function depicted in Table 1 is based on the following observations:

1. Layer I — The intuition is that, given the initial track, a minimal plan would determine which targets are accessible for intercept. The output is a minimal Weapon Tasking Plan (WTP) using autonomous interceptions.
2. Layer II — Additional complexity is associated with consideration of background effects for the success of an intercept. Consideration may be given either to modular enhancement of the Layer I function or to implementation of a diverse algorithm.
3. Layer III — An In Flight Target Update (IFTU) adds additional planning complexity that results in improving the probability of intercept. This complexity involves manipulation of sensors and weapons to acquire the data for the

IFTU as well as the communications (WTP, STP and CTP) for transmission and receipt.

4. Layer IV — Near-optimal battle planning is facilitated by coordination of all system elements. Further computational and system complexity is introduced.

6 SUMMARY AND RECOMMENDATIONS

This paper represents an initial study in complexity management. We have developed a generic model for representing the GPALS BM/C^3 system. Our approach and modeling paradigm have the following characteristics:

1. The GPALS system comprises three critical functions: Sense, Plan, and Execute.
2. Software implementations of functions may be replicated across many system elements.
3. Therefore, single instantiations of a BM function may not be critical if a *complexity management strategy* is implemented.
4. Complexity management facilitates graceful degradation of a computer-intensive BM/C^3 element.
5. Trusted computer resources are an enabling capability for such a design.

We have identified the following topics for future research and engineering directions:

1. Determination of the critical functions for the trusted computer resources.
2. Experimentation to support development of the alternative concepts for firewalls.
3. Demonstration of the multilayered design paradigm. This would be performed in consort with work on the trusted computer resource.
4. Development of a methodical approach for application of a Trusted Complexity Management Strategy.
5. A case study into the application of diverse formal methods to the design of representative, critical pieces of SDIO-GPALS BM/C^3.

ACKNOWLEDGEMENTS

This report is the result of efforts by several efforts. The first effort was provided by a group of individuals that presented briefings on specific BM architectures. All the briefings provided extremely useful insights into the critical functions of BM systems. Thanks are extended to Paul Dayton (General Electric — Aerospace Division), Bob Kinney (Sparta and AIS), and Phil Dunlap and Dick Kitus (Rockwell International — Strategic Defense Center) for providing us with briefings on BM architectures.

The second effort was expended by members of the critical functions task group. Thanks are extended to these individuals for attending multiple meetings and for lending their expertise to the development of the report: David Castañon, Boston University; Danny Holtzman, Vanguard Research, Inc.; Richard Kemmerer, The University of California – Santa Barbara; Jay Lala, The Charles Stark Draper Laboratory; John McHugh, The University of North Carolina; John Rushby, SRI International; and Brian Smith, The University of New Mexico.

A third effort was conducted by the Argonne National Laboratory technical and support staff. This effort involved attending the briefings and participating in numerous debriefings. This group includes Mark Henderson, Gail Piper, and Jim Boyle. Mark Henderson spent many hours of useful discussions, and Gail developed an initial report from a set of overhead slides.

We thank Lt. Col. James Sweeder, Carol Friedhoffer, and Carol Taylor. This study was performed under the auspices of the SDIO, under PMA 2304, and of the NSA, under MOD 708992.

Bibliography

[1] E. W. Dijkstra, "The humble programmer," in *ACM Turing Award Lectures*, pp. 17–31, New York, New York: ACM Press, 1972.

[2] GE Aerospace, Strategic Systems Department, P.O. Box 1000/ Building 98, Blue Bell, PA 19422, *Trusted Software Development Methodology*, CDRL A075 — POC Version ed., July 1991.

Appendix A Validation of a Generic BM/C^3 Model

Section 4 of this report describes a generic BM/C^3 model. This appendix provides a description of the three architectures from which this model was derived.

Modeling Three Architectures

Figures A.1–A.9 depict the three BM/C^3 architectures for the Sense, Plan, and Execute functions, respectively. The close correlation between these models provides an informal validation of our generic model. The following sections provide a description of each architecture.

The level 2 Sense function is allocated to all sensors within the respective architectures.

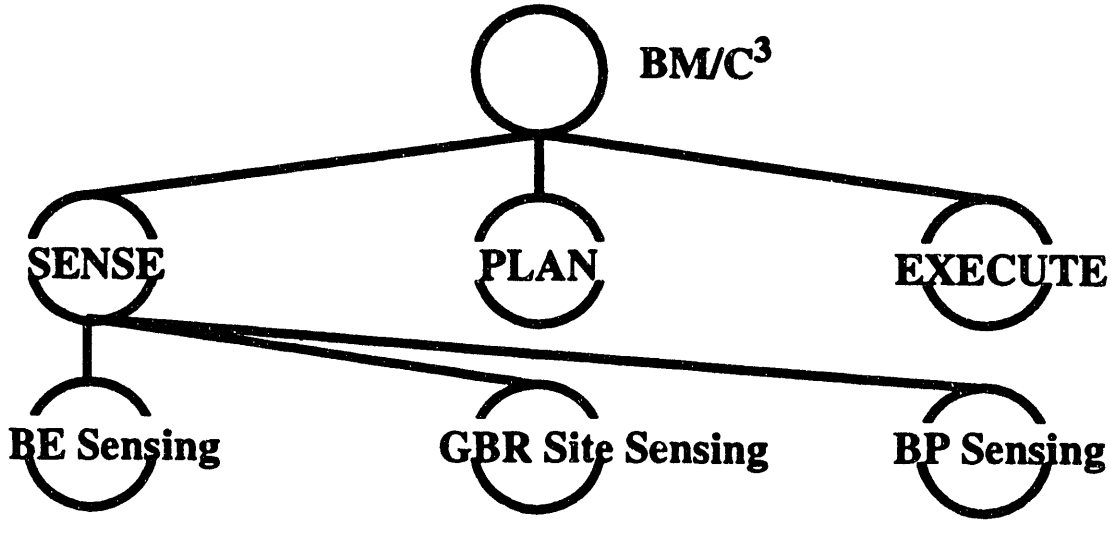


Figure A.1 GPALS Architecture #1 – Level 2 – Sense Function

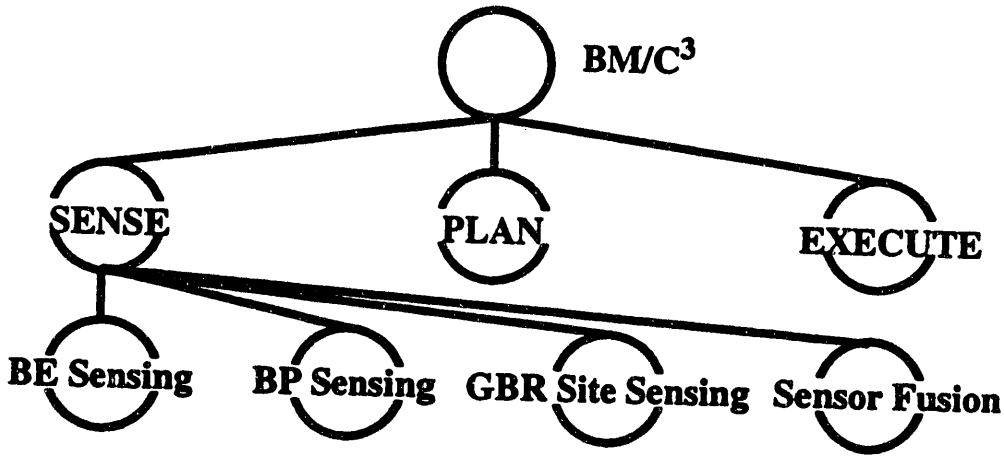


Figure A.2 GPALS Architecture #2 – Level 2 – Sense Function

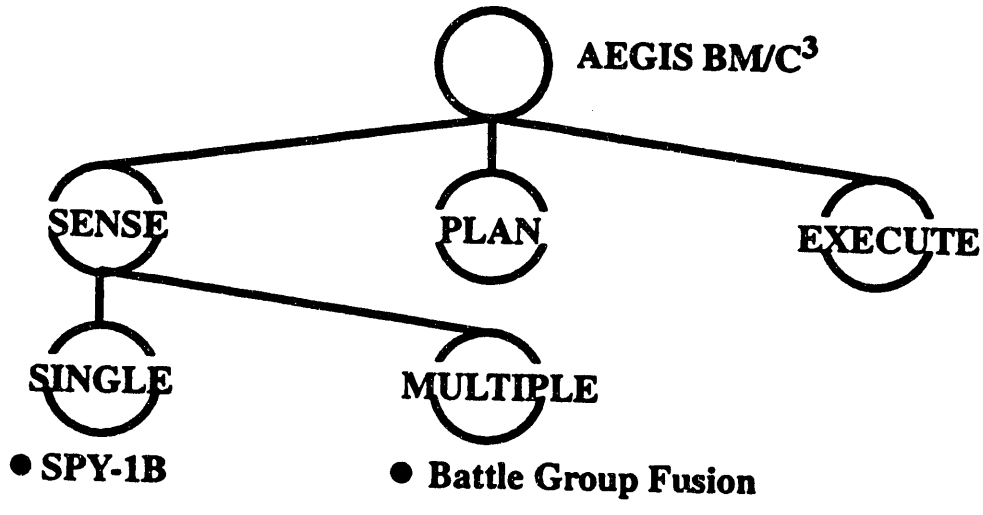


Figure A.3 AEGIS Architecture – Level 2 – Sense Function

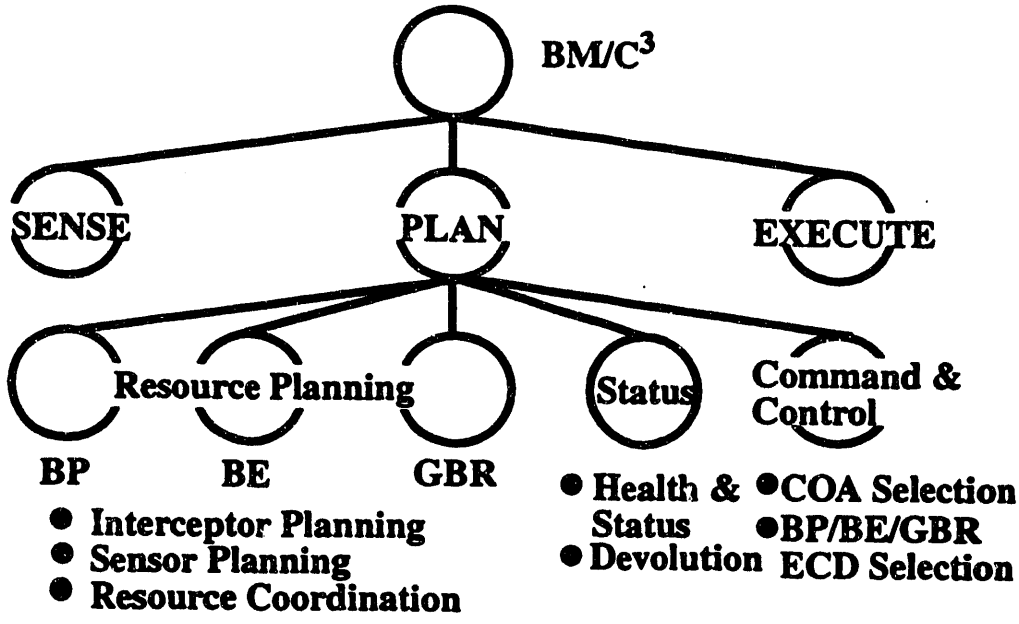


Figure A.4 GPALS Architecture #1 - Level 2 - Plan Function

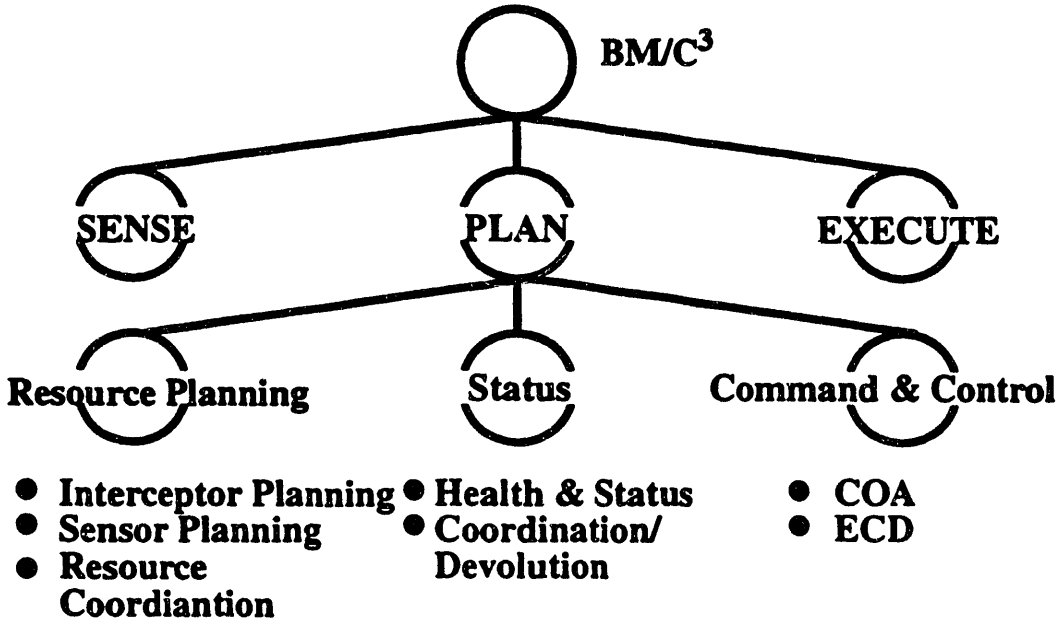


Figure A.5 GPALS Architecture #2 - Level 2 - Plan Function

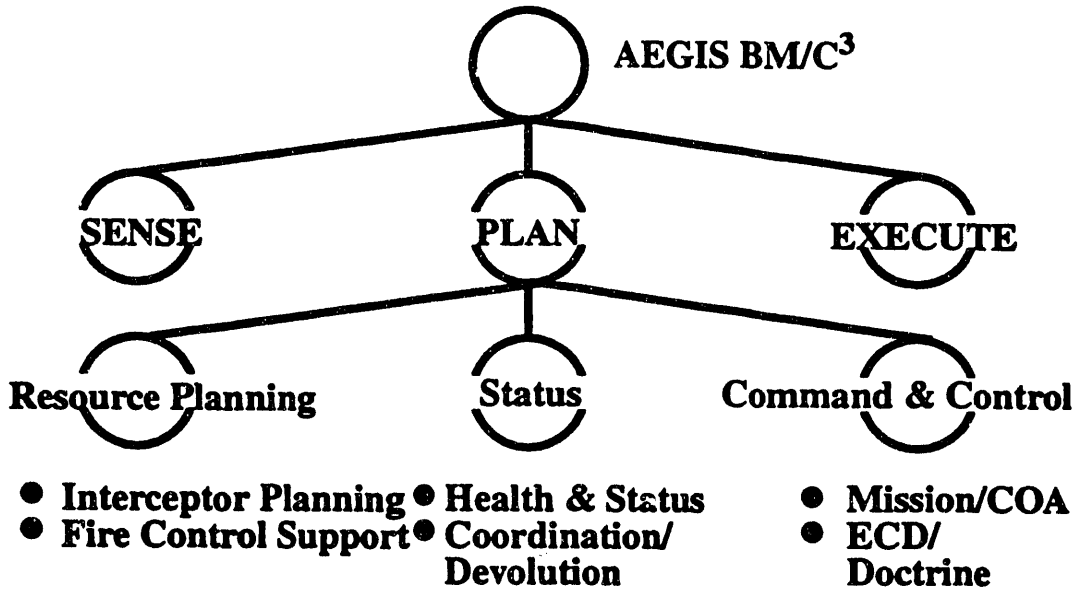


Figure A.6 AEGIS Architecture – Level 2 – Plan Function

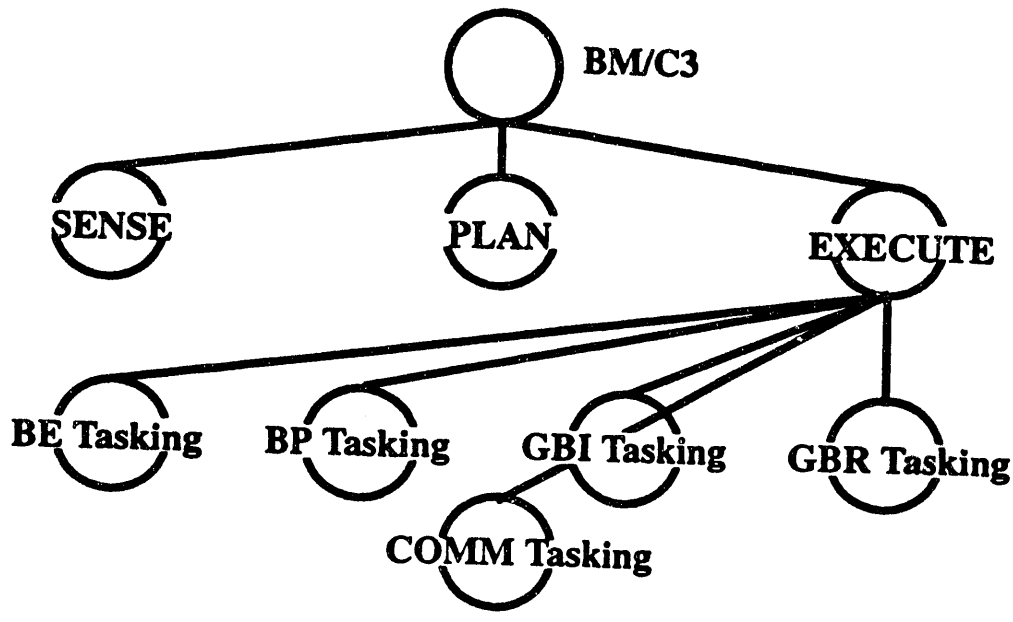


Figure A.7 GPALS Architecture #1 – Level 2 – Execute Function

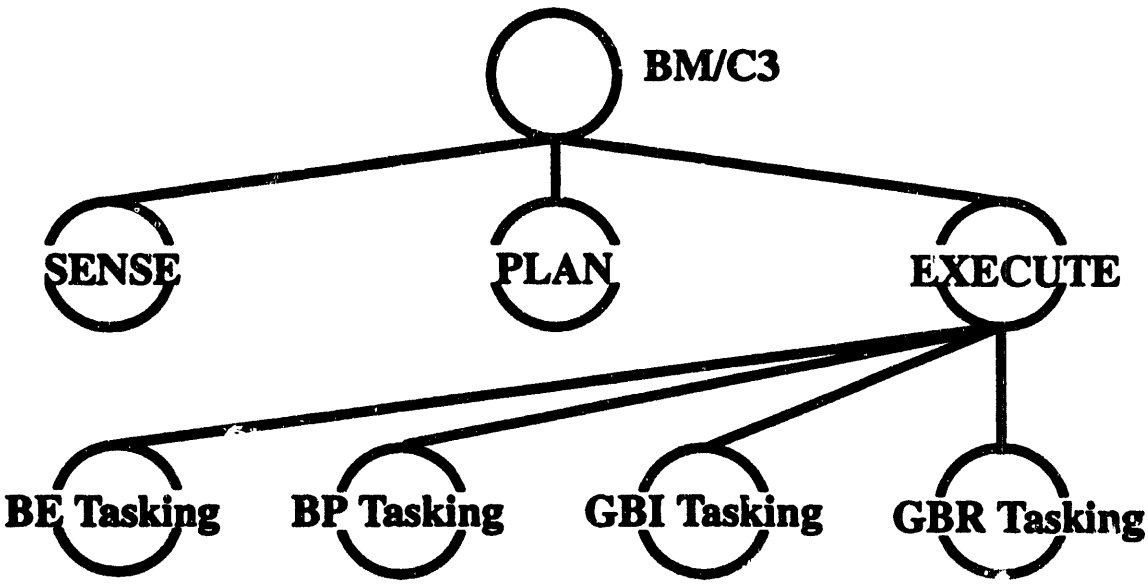


Figure A.8 GPALS Architecture #2 – Level 2 – Execute Function

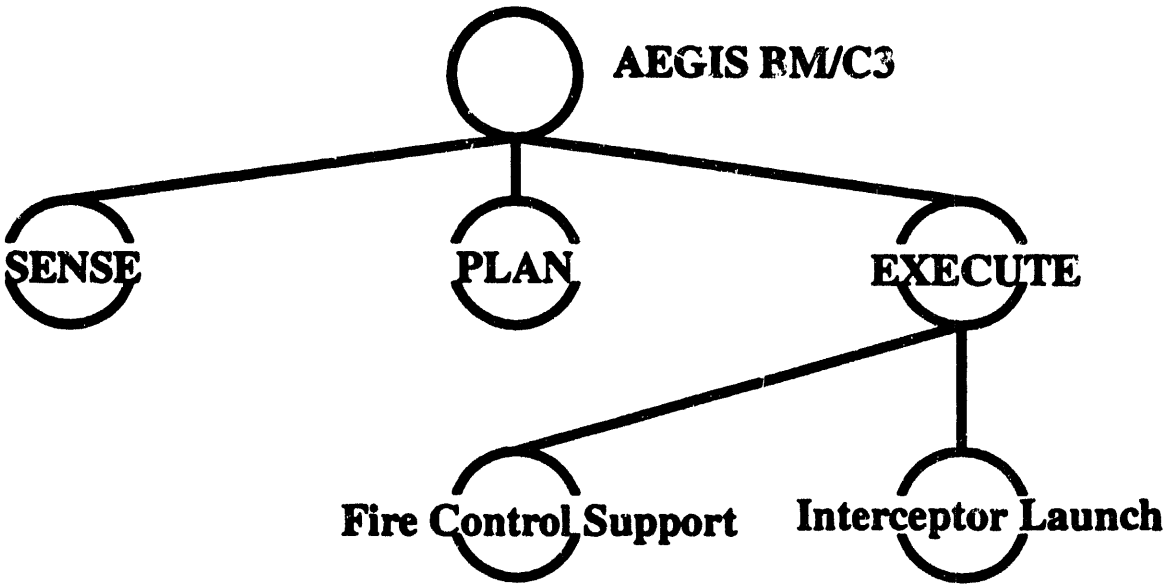


Figure A.9 AEGIS Architecture – Level 2 – Execute Function

Table A.2 Three Degradation Strategies

Strategy	Implication	Critical Function	Strength
A. *Failures* resulting in isolation or disabling of one or more sites	Divide battle space into tiers, independent elements	Communication assured at element level, diverse BM's	Space and Land SPE Triads or Elements
B. *Failures* in communications network	Flood routing of data, redundant BM's	BM's	Flexible SPE triads, multiple modes of operation
C. Multiple *Failure* scenarios	All scenarios accomodated	Manual operation	Flexibility

Degradation Strategies and the Battle Manager

Implicit restructuring of the system occurs during both the planned devolvements as a battle progresses and the unplanned devolvements from the failure of system components during a battle. Usually, this devolvement results in a downsizing of an operational unit, possibly accompanied by the fragmentation of an operational unit and the transfer of operational authority into the reduced unit.

As an example of a planned devolvement, consider the case of an intercepter entering the final stages of its operation upon receipt of an In-Flight Target Update (IFTU). If everything goes as planned, the IFTU is delivered to the vehicle, which then uses the updated target information in combination with data from its own sensors to perform the maneuvering necessary to hit the target. During the final stages, the Sense-Plan-Execute (SPE) triad is contained entirely within the interceptor. The planned devolvement occurs when planning execution authority is transferred from the external control source providing the IFTU to the interceptors internal mechanisms. In the event that the IFTU is not received, the devolvement still occurs on a pre-planned basis, and the interceptor proceeds on the strength of its own observations and the target information provided at launch time.

The SPE triad in an autonomous interceptor in its final phase of operation is the least complex and most compact of those involved in the GPALS system. It comes into existence at the point at which the interceptor assumes control over its

own operations. At this point, the interceptor has a very simple mission statement: intercept and destroy a particular target. For this mission to succeed, the GPALS system must have executed a plan that positioned the interceptor within range of the target, positioned so that the target can be sensed and the interceptor maneuvered to it. A high-level formal specification for this functionality should be straightforward. The proper positioning becomes a precondition; the terminal sensing, planning and execution operations constitute the specification body.

Admittedly, the refinement of this specification to cover an actual implementation and realization of terminal guidance for a particular interceptor is not trivial. Whether done by formal methods or other means, assuring the terminal interceptor functionality is a necessary part of assuming proper system operation.

As we work backwards from the terminal interceptor SPE triad, we find that the triad required to position the interceptor within the feasible interception region is somewhat more complicated. From the time that the interceptor is launched until the time that its IFTU is transmitted, portions of the planning function reside in a number of locations. Some planning is being done to ensure that the appropriate facilities are available for transmitting the IFTU. Some planning is involved in accumulating the data for the updated target map and in computing it at the appropriate time. Additional planning is required for operating the sensors to acquire the targets. Depending on the way in which the interceptor works, it may have an internal planning function operating during the flyout.

At this time, some portion of the GPALS global sensing function, involving one or more sensors and phenomena is devoted to providing sensor data for the target update of the interceptors target map. The interceptor may have a local sense function for guidance during flyout. The flyout phase ends with the execution of the IFTU, which may require actions at a number of locations.

Figure A.10 depicts the terminal SPE Triad in the context of the Generic BM/C^3 model.

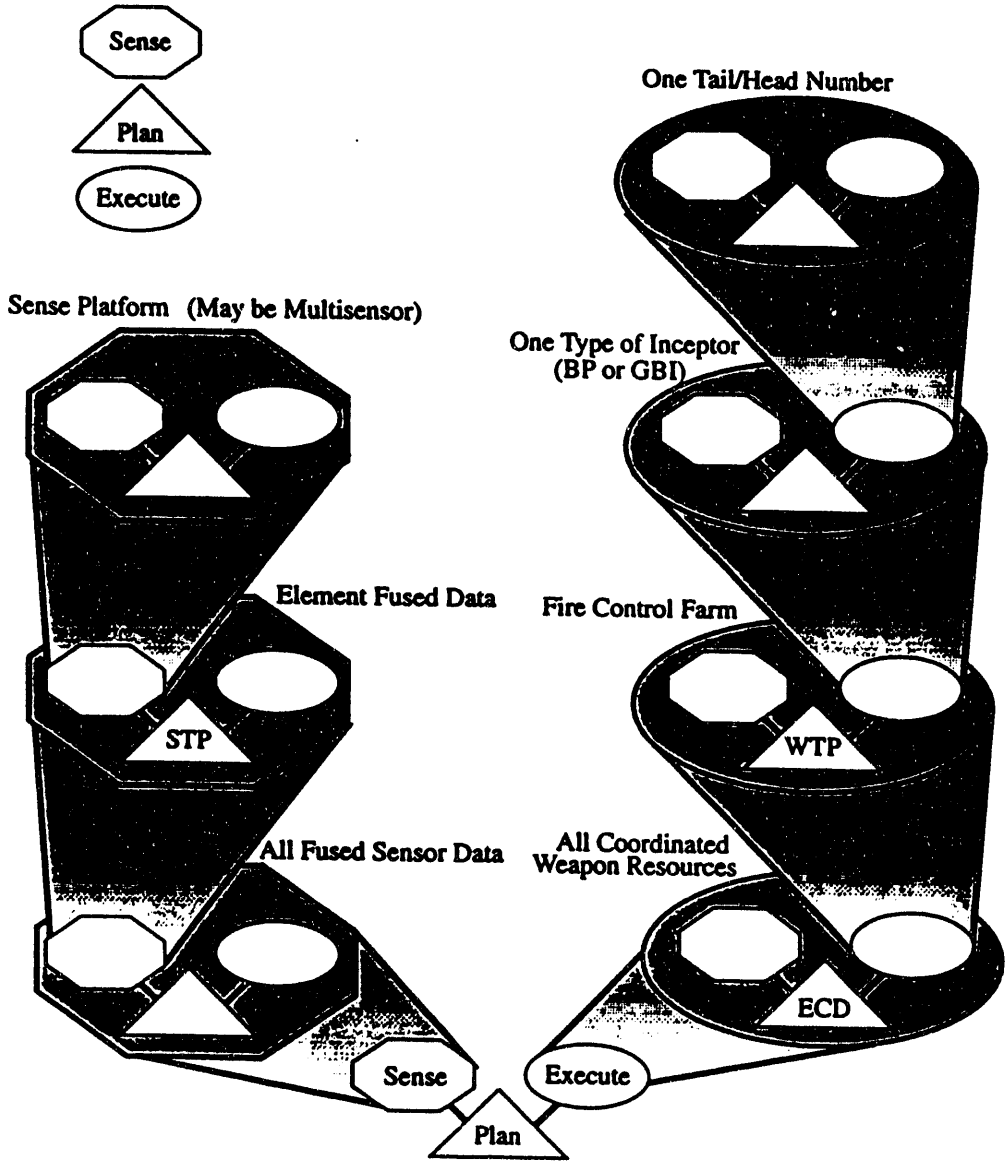


Figure A.10 The SPE Triad — a repeating pattern within the generic model

Appendix B AIS Architecture Summary

Jay Lala
Fault Tolerant Systems Division
The Charles Stark Draper Laboratory
Cambridge, MA 02139
and
John Rushby
Computer Science Laboratory
SRI International
Menlo Park, CA 94025

Architecture Overview

The AIS philosophy and approach are different from those of any other major defense procurement. With changing threats, mission, and budget, but deployment nonetheless mandated for 1996, the AIS group felt that the usual DoD Waterfall procurement process was inappropriate. Instead, they advocate, and have used, a process that iterates on all of the BM/C3 parameters. The process begins with the Mission Objectives (preserve the American Way Of Life) and Geopolitical Futures, converts these to effectiveness and performance requirements, and, after iterating on concepts/design features, functional allocations, cost/schedule, and component trades, ends up with a preferred architecture/program strategy. Because of this approach, the AIS architecture is flexible in many details, but it is recognized that BM/C3 is the "glue" that ties it all together and that allows an evolving system to be deployed.

Development of a BM/C3 architecture was hampered by absence of Battle Management as an explicit element in SDIO for several years. One of the early accomplishments of AIS was to get this corrected. However, development of the BM/C3 architecture is 1–2 years behind that of the GPALS hardware (they are ready to build interceptors), and it is seen as the "long pole in the tent." A significant influence on the AIS architecture was a paper entitled BM/C3 by Danny Cohen and others.

The AIS architecture recognizes that there are two classes of requirements:

1. Explicit requirements, overtly expressed in the mission statement, and
2. Implicit requirements that the customer expects to be met, without explicitly stating them.

For example, wanting an air conditioner in a car is an explicit requirement; not having doors fall off is an implicit requirement. Implicit requirements are particularly difficult, because they tend to get overlooked in the contracting process (no contractor is going to deliver anything that is not specified). AIS did an exercise to identify and make explicit some of the implicit requirements. These include a low false alarm rate and the desire not to fire at the space shuttle.

The explicit requirements were identified as the following. Note that all these are primarily the responsibility of BM/C3:

1. Leakage of less than 1% (per RV) against a limited strike (a higher leakage is expected and accepted against larger, non-GPALS, threats),
2. Minimized inventory (less threatening to the CIS and the US taxpayer),
3. Flexibility to evolve (with respect to threat, GPALS elements, deployment sequence), and
4. Capable of evolution to counter a resurgent (in sophistication as well as numbers) threat.

After much debate, AIS identified 5 or 6 major "features":

1. multiple, user-selectable, modes of BM/C3 operation,
2. multiple, hierarchical, levels of human control (adaptable to different CINCs),
3. multiple, user-selectable engagement tactics,
4. adaptable "flood routing" of all available data when practicable,
5. modular BM/C3 node design and implementations, and
6. a focussed, pragmatic, approach to the program.

The exact presentation and characterization of these items are evolving, but two are of paramount importance: the flood-routing of data, and multisensor fusion. The latter does not show up explicitly in the list above, but that is a consequence of the evolving nature of the characterization. Also, note that while all the other items listed are concerned with policy, the "flood-routing" item is a mechanism. The purpose of this mechanism is to ensure flexibility, resilience, and fault-tolerance.

Architecture

The main feature of the AIS architecture is that it includes a number (up to about 15) identical BM/C3 nodes, any of which is capable of running the entire battle. Individual sensor data is retrieved and reduced (probably to tracks) by its own processing element (which may be in space, or on the ground, or a combination of both), and is then logically broadcast (the network will be mostly point to point rather than a physical broadcast medium) to all BM/C3 nodes. (This is the "flood-routing" of data mentioned earlier. The amount of data generated is small: estimated at no more than 600 kbytes/sec across the whole system.) Each BM/C3 node will fuse the arriving data with other data in a process called multi-sensor fusion (some single-phenomenology fusion may take place local to the sensors), and constructs plans periodically (say, every 10 seconds). The plans developed at individual nodes are communicated and compared. One interpretation is that the node that is in "control" communicates its plan to all the others; another interpretation is that all nodes broadcast their plans to all others, just like the flood-routing of processed sensor data. Individual plans will be different, since some nodes will have missed certain data, or may have received it in different orders. But it is hoped that they will be "close" in some sense.

The architecture accommodates a range of operating modes from fully coordinated through semi-autonomous to fully autonomous. In the coordinated mode (which is the initial, and most desired mode of operation), a single node is in charge, and its plan is executed (in a distributed fashion, by the other BM/C3 nodes). The other nodes monitor the leader's plan, however, and can outvote it if necessary. If the leader dies or gets outvoted, either another leader is elected, or the system devolves to a semi-autonomous mode in which resources are partitioned into a few "enclaves" that coordinate internally as described above, but separate enclaves do not coordinate with each other. In the fully autonomous mode, each BM/C3 node operates on its own to the best of its ability.

The benefits of this architecture are as follows:

1. It uses as much sensor data as possible (maximizing confidence),
2. It provides fully coordinated operation when possible, but
3. It avoids fragility inherent in total reliance on full coordination.
4. It is a similar scheme to that envisaged for "Brilliant Pebbles."
5. It permits multisensor tasking (load-sharing can be desirable).

6. It does not rely on perfect knowledge of the threats that may be faced.
7. It does not require perfect detection or identification of targets by any one element.

Single-point (common mode) failure is a possibility with BM/C3 node replication, but believes these can be mitigated using state of the art techniques, including: hardware; distributed systems, voting, and software; independent developers, IV&V, independent DT&E, multiple parallel systems, OT&E, etc.

Ultimately, the measures taken to guard against single-point failure will depend on the resources available.

Analysis

The flood-routing concept, while appealing in the abstract, may raise a number of difficult coordination and fault-tolerance problems when considered in concrete detail. This is not to say that the concept is not a good one, but only that it does not, of itself, solve the challenges of coordination, resilience, and fault tolerance. For example, the data that is flood-routed is not raw sensor data, but processed data (e.g., tracks). If the data reduction for a given sensor is done at a single site, then one has a single point of failure for that sensor (other than the sensor itself); if it can be done at multiple sites, then one has a problem of coordination and allocation among those sites; if it is done in parallel at multiple sites, then one has a problem of consistency and agreement (and a lot of data moving about).

One problem with this approach is that a single point or common mode failure might not simply reduce effectiveness, but could prevent the system from working at all the one time it is needed. A second problem is that resilience is not necessarily cumulative: adding redundant components and alternative mechanisms increases complexity, and with it the likelihood of system-level design faults. It would be preferred to eliminate single-point failures — that is, a system that is guaranteed (to some specified degree) to work, and whose effectiveness is governed by the available budget, rather than one that will have high effectiveness if it works, and for which the assurance that it will work is governed by the available budget.

Distribution for ANL-93/1

Internal:

J. M. Beumer (100)
J. M. Boyle
A. P. Campbell
G. H. Chisholm (100)
H. Drucker
F. Y. Fradin
M. W. Henderson
G. W. Pieper
L Russell
R. L. Stevens
T. G. Surles
C. L. Wilkinson

ANL Patent Department
TIS File

External:

DOE-OSTI (12)
ANL-E Library
ANL-W Library
Manager, Chicago Field Office, DOE
Mathematics and Computer Science Division Review Committee:
W. W. Bledsoe, The University of Texas, Austin
B. L. Buzbee, National Center for Atmospheric Research
J. G. Glimm, State University of New York at Stony Brook
M. T. Heath, University of Illinois, Urbana
E. F. Infante, University of Minnesota
D. O'Leary, University of Maryland
R. E. O'Malley, Rensselaer Polytechnic Institute
M. H. Schultz, Yale University
D. Castanon, Boston University
J. Cavallini, Department of Energy - Energy Research
P. Dayton, General Electric Co., Aerospace Division
P. Dunlap, Rockwell International, Strategic Defense Center
C. Friedhoffer, National Security Agency
D. Holtzman, Vanguard Research, Inc.
F. Howes, Department of Energy - Energy Research
R. Kemmerer, The University of California at Santa Barbara
R. Kinney, Sparta and AIS
R. Kitrus, Rockwell International, Strategic Defense Center
J. Lala, The Charles Stark Draper Laboratory
J. McHugh, The University of North Carolina
J. Rushby, SRI International
B. T. Smith, The University of New Mexico
Lt. Col. J. Sweeder, U.S. Dept. of Defense
C. Taylor, National Security Agency

END

DATE FILMED

3/29/93