Summer 2005

A Conceptual Framework for Analysis of System Safety Interoperability of United States Navy's Combat Systems

Showkat Shanaz Alborzi
Old Dominion University

A CONCEPTUAL FRAMEWORK FOR ANALYSIS OF SYSTEM SAFETY INTEROPERABILITY OF U.S. NAVY'S COMBAT SYSTEMS

by

Showkat Shanaz Alborzi
B.A., Computer Science and Applied Statistics, December 1989,
St. Mary University
M.E., Systems Engineering, May 2001, University of Virginia

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirement for the degree of

DOCTOR OF PHILOSOPHY

ENGINEERING MANGEMENT and SYSTEMS ENGINEERING

OLD DOMINION UNIVERSITY
August 2005

Approved by:

_____
Ji Hyon Mun (Advisor)

_____
Resit Unal (member)

_____
C. Ariel Pinto (member)

_____
Tom English (member)

ABSTRACT

A CONCEPTUAL FRAMEWORK FOR ANALYSIS OF SYSTEM SAFETY INTEROPERABILITY OF U.S. NAVY'S COMBAT SYSTEMS

Showkat Shanaz Alborzi
Old Dominion University, 2005
Advisor: Dr. Ji Hyon Mun

Today's political and military reality requires the optimal use of our legacy systems. The objective is to maximize the effectiveness of our operations by efficient allocation, placement and the use of our forces and war-fighting systems. The synergism drawn from the capabilities of the legacy complex systems enables today's war-fighting needs to be met without substantial increase in cost or resources. This synergism can be realized by the effective integration and interoperation of legacy systems into a larger, more complex system of systems.

However, the independently developed legacy systems in this new tactical environment often have different data types, languages, data modeling, operating systems, etc. These differences are impediments to the requirement for interoperability, and can create an environment of confusion, misinformation and certainly un-interoperability, hence hinder the safe interoperation of the meta-system and potentially increase the risk for mishaps. Safe interoperability capability assures that the mission objectives are achieved not only effectively but also safely.

The System Safety Interoperability Framework (SSIF) introduced in this dissertation provides the framework for the engineering community to evaluate, from system safety perspective, the interoperability issues between multiple complex systems in the U.S. Navy's system of systems context. SSIF characterization attributes are System of Systems (SoS) tactical environment, SoS Engineering, SoS Safety Engineering, and Safety Critical Data. SSIF is applied to AEGIS Ballistic Missile Defense 3.0 Program to explore and analyze the safety interoperability issues in the overall system, by which the SSIF is further validated as an effective approach in analyzing the safe interoperability capability in Navy's combat systems.

This dissertation is dedicated

to

My Lord and Savior Jesus Christ
whose cross and passion I carry in this life, whose light I hang in my heart, and whose peace I keep in my soul

for,

"I bear, on my body, the marks of Jesus"

ACKNOWLEDGEMENTS

I would like to, first and foremost, thank my advisor Dr. Nina Mun for her guidance, efforts, and assistance throughout my last year of study. Dr. Mun's commitment, responsiveness, and genuine interest in my work, truly accounts for the success of this dissertation. I thank my committee members- Dr. Resit Unal, Dr. Ariel Pinto, and Dr. Tom English for all their valuable feedback, support, and time spent on reviewing my work.

My sincere appreciation to Gary Friedman, the Aegis Ballistic Missile Defense Principal for Safety, for his support, contributions to the safety interoperability definition and SSIF review, and his assistance in getting the U.S. approval for release. I thank Kevin Stottlar, the Aegis Principal for Safety, for his support, insights, and the review of SSIF. I thank Doug Bower for his review of SSIF and his continuous support. Many thanks to Mike Brown for his contribution of examples of interoperability conflicts, and sharing of his expertise.

I extend my gratitude to the NAVSEA sponsors, to June Drake for getting the U.S. government release, and to my kind and supportive managers, Mike Till and Brad Cobb. Special thanks to Mike Ramsburg from "Dr. Showkitty", for his continuous encouragements!

I'm deeply thankful to all my dear friends at the Covenant Community Church who, with their love and prayers, walked step by step with me throughout my final year. Special thanks to my dear Gail who is smiling at me from heaven. Her love and sweet memory will forever be in my heart. Words of gratitude are toward Martha Flores for her unwavering encouragements and support. My deepest thanks to Lisa and Greg Miller for their love.

To the one who left his footprints in my heart, a 'general comment'- your memory will be a hidden treasure in the depth of my heart; neither can it 'fade away' nor get lost in the chaos of human iniquities. Thank you for the experience of knowing you and the privilege of loving you.

At last, my special thanks to my life's most precious blessings, Abraham, Rebecca, and my sweet Lily for filling my life with love, joy, and pride. Being your mom has been the greatest honor bestowed on me by God. Thank you.

TABLE OF CONTENT

Chapter **Page**

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

CHAPTER I

INTRODUCTION

"It is immoral to design a product or system for mankind without recognition and evaluation of the hazards associated with that product or system."

-Anonymous safety engineer

1.1 NEED FOR SYNERGY

The U.S. Navy's war-fighting systems are increasingly more complex and interrelated than ever before. To make sense of the chaos created by complexity, communications, dependencies, etc., we need to advance our systems thinking. We need to find the patterns in complexities that can be understood, simplified and controlled.

Today's political climate often dictates employing our forces in the most operationally optimized order, often referred to as "The Order of Battle". This employment can be planned sometime in advance or it can be a sudden employment by a presidential directive. The objective remains the same-- to maximize the effectiveness of our operations by efficient allocation, placement and the use of our forces, war-fighting systems and facilities.

To achieve this challenging new requirement, Department of Defense (DoD) has initiated a new strategy to apply synergism to existing capabilities (Alborzi, 2004). The synergism is achieved by integrating the legacy systems into a larger, more complex and more capable system of systems.

The higher level of operational requirements is achieved by using the functional capabilities of each system to work together for even greater effect and outcome in operations. This initiative requires the complex systems to interoperate and perform joint task operations. The requirement for this interoperation and performance of joint tasks is achieved by using Commercial-Off-The-Shelves (COTS) and Government Off-The-Shelves (GOTS) software. The reliability and safety of the software is a major factor in the overall effectiveness of tactical operations. Impediments to safe operation of the systems will increase the risk for mishaps. Safe interoperability capability ensures that the mission objectives are achieved not only effectively but also safely.

1.2 PURPOSE

The main purpose of this research is to develop and apply a framework to analyze the system safety interoperability of the Navy's complex war-fighting systems of systems. War-fighting systems include weapons systems, command and control (C2) systems, communication systems, radars, planner systems, display systems, ship self-defense systems, etc.

The journal model for the references herein is *The American Psychologist*, the journal of the American Psychological Association

1.3 MOTIVATION

The factors below are the major motivators behind this research.

1.3.1 Mishap Reduction

The Center for Army Lessons Learned reports (CALL, 2003):

> 17 April 2002; four Canadian soldiers are killed and eight wounded when a USAF F-16 mistakenly bombed them as the Canadians were engaging in a live-fire training exercise at night.
>
> 24 March 2003; a Patriot Missile Battery destroyed a British Tornado returning from a mission in Iraq. Two British pilots perish.
>
> 4 December 2001; three Special Forces soldiers and seven Afghanis were killed and many more injured when they mistakenly called in a air strike by a USAF B-52 on their own position.
>
> 6 April 2003; an USAF A-10 kills one British soldier and injures several in a friendly fire incident in southern Iraq.

The real life examples above show that the cost of conflicts both in terms of dollars and human lives is so high that it is imperative to rethink our planning strategy to include meeting the operational objectives with minimal cost and weaponry and minimal loss of lives. Hence, the requirement for interoperability of our systems and forces must include safety study, safety assessment and risk mitigation in the system of systems (SoS) environment.

In theory, any tactical operation can eventually be won if we were to have endless time for fighting, unlimited budget, endless numbers and types of weaponry, and without any consideration for the loss of lives, friendly fires, loss of assets or missiles. In reality, however, the DoD faces budget cuts, manpower issues and has to do more with less. It needs to achieve the operational objectives with minimal cost and weaponry, minimal loss of lives, and in the shortest possible time. This requires the planners to develop the mapping of the Order of Battle with carefully selected participating systems for deployment. As a part of this selection, not only the participating systems must be able to integrate, coordinate, and exchange data, but they must also be able to interoperate in a safe manner.

Given the enormous impact that just one friendly fire might have in terms of loss of lives and the associated liabilities associated, not to mention its impact on the stand of the country in the global political context and its impact on the overall economy, reducing potential mishaps by identifying, analyzing and mitigating the issues of safety interoperability in the U.S. Navy's complex systems is not only a good engineering practice, a technical necessity, or even a moral obligation, but a wise and prudent decision in terms of associated liabilities.

1.3.2 Designing Safe Systems

The second motivator for conducting this research is to learn how to design safety into the system of systems so that it becomes the "quality" or "property" of the system. The Office of the Under Secretary of Defense's Defense Acquisition Desk Book (DoD, 2002) calls out to DoD Program Managers to, "...be aware the process of designing safety into systems directly effects operational safety." Clearly, this 'linkage' not only lays the foundation for a need for system safety analyses, but, with directly linking it to operational safety, it implies that less 'accidents' in the operational phase will be the end result when safety is designed into the system. The handbook further adds that the Program Manager is, "to make safety a priority in system design" (DoD, 2002). The importance of making "safety" a priority in system design is often overlooked usually because of lack of recognition that safety is a property of a system not of the components of a system, and most likely because of funding constraints. Just as we design functional capabilities into the system so it could perform its intended mission (and that becomes the property of the system), we must design safety into the system, and thus it becomes the property of the system so it could perform its intended mission in a safe manner.

When we integrate heterogeneous systems (as in the case of SoS architecture), we change the system's context, and by doing so, the safety properties change. When systems are integrated, safety attributes (safeguards, controls, etc.) of the components of that system may be circumvented, overridden or negated, or at minimum affected. It is this change and the follow-up uncertainties in the overall behavior of the federation that requires us to view the "health" of the system from safety perspective, holistically, systemically and within its new context. This holistic system safety study will enable the safety engineer and the development engineers to be aware of the dependencies between safety-related software artifacts and their association with the changes.

1.3.3 Understanding Software Safety

The author was also motivated by the need for "understanding" software safety. If we can understand the software safety, its attributes, its criteria for criticality, its impact on the system operations, then we can improve and elevate the safety quality of the system.

Understanding the software of a complex system is already an extremely difficult task. It requires special skills, a set of critical eyes and years of experience. To understand software safety, it requires all of the above, plus going beyond the given domain, beyond the validated limits, and with "safety glasses" firmly on.

Safety critical functions of the software must be understood fully in order to implement controls for risk mitigation. Software errors, dead code, run-time errors, timing anomalies in safety critical parts of the software could introduce a hazardous condition that may contribute to a mishap, such as an inadvertent

launch or restrain firing (misfire). The Defense Acquisition Deskbook states that software needs to be analyzed to "identify potential system hazards contributed to by the software or software environment and to examine the causal factors so as to eliminate or mitigate the hazard risks throughout the continued development of the software" (DoD, 2002). Moreover, the task of understanding the software becomes even more crucial when we integrate COTS software among our legacy software in the SoS environment.

The challenge starts with various computer languages used in different systems. AEGIS baselines are written in CMS-2L, a Navy-unique language, and ADA. The Advanced Tomahawk Weapon Control Systems (ATWCS) is written in ADA-98, with some C. Understanding the software with safety impact in an integrated system of systems with diverse languages, terminologies and requirements is crucial in reducing the risks of mishaps during tactical operations.

1.3.4 Safety Risk Reduction /Management

The fourth motivator is to learn how to reduce and manage safety risks associated with the software. Risk taking is a part of the reality of everyday work, mainly because risk is an inherent property of today's complex systems. A "safe" car is a car that is always parked in a garage (never driven.) A totally safe missile launch is not launching at all. Truly "safe" software is software than doesn't "talk" to the outside world, so it doesn't "make" others do something.
Now that we know we have to "live" with risk, we are obligated to understand the risks (i.e. to assess), to design automated safeguards/interlocks (i.e. to reduce), and to control them (i.e. to manage.) Although, system safety as a whole is considered a risk reduction approach for early identification, analysis, elimination, and/or control of hazards (DoD, 2002), a holistic systems safety analysis will help us to find ways to reduce and manage risks in the new, higher level, more complex system of systems.

A framework whereas system safety engineers can use to apply engineering tools and techniques in analysis of safety critical data is necessary and is the focus of this research.

1.4 THE RESEARCH PROBLEM

The current reality in Department of Defense (DoD) is that we need to do more with less due to the budget constraints. This means optimizing the use of legacy tactical systems that, at one time, were developed to perform as single systems with no integration or interoperability requirements. Now these heterogeneous systems need to integrate and interoperate to provide seamless operational objectives. This higher level of integration of multi-complex systems introduces an emergent issue- safety. The problem is that although few engineers in Department of the Navy have taken it upon themselves to conduct an integrated effort to analyze the system of systems for safety, there is no requirement,

mandate or even a guiding standard to do so. This research attempts to identify and evaluate the safety issues in interoperability of complex combat systems so that the mission requirements can be achieved safely as well as effectively.

1.4.1 The Research Questions / Objectives

- What is system safety interoperability for the Navy's complex combat systems?
- How can safety interoperability be analyzed for a Navy's complex combat system?

This research has been conducted to meet the objectives listed above by using a real life complex system of systems, namely AEGIS Ballistic Missile Defense (BMD) 3.0 system for analyses.

1.5 ASSUMPTIONS/ LIMITATIONS

1. The study will use only UNCLASSIFIED information.
2. The study will not attempt to resolve the safety interoperability issues.
3. The study will be limited to U.S. Navy combat systems.
4. The study will assume that complex subsystems have been through single-system safety engineering programs.
5. This study assumes that interoperability capability is part of the design requirements.
6. This study will not attempt to achieve interoperability.
7. This study will be limited to software driven issue in safety related interoperability.

Deviation from the Proposed Plan

Based on the changes to the overall program plan of AEGIS BMD development (development schedule moved to the right two months), this study will use the AEGIS BMD 3.0 computer program for application of SSIF and the analysis of the system. This small change will not affect the outcome or expectations or quality of the analysis.

1.6 DEFINITIONS / DESCRIPTIONS

System

System is defined as, "A set of components that act together as a whole to achieve a common goal" (Leveson, 1999). The components are interrelated and may have direct or indirect interfaces to other components. A system is part of the environment that it will operate. System boundary is defined and input to and output from the system is explicitly known. A system may contain subsystems and may be a part of a larger system, i.e. system of systems.

Stephans (Stephans, 2004) depicts the system as:

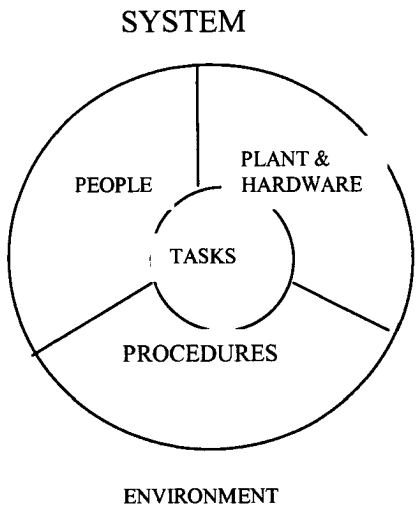

Figure 1.1 - System Definition

A system can be a 'simple" system such as a clock with a few or no input, an output, a limited or no human interaction, no decision-making capability, and a limited or no interfaces to the outside world.

A system can be "complex" when the parts or components are interrelated and the behavior of the parts affect the overall system both negatively or positively. More discussion regarding complexity provided in "complex systems" below.

Safety

Every complex system has some inherent hazards by the nature of its existence that poses risk for its users. An automobile runs on fuel to perform its function—driving. Fuel being flammable in nature, is a potential source of fire that can lead to burning the car, killing the driver and its passengers. The probability of occurrence of the mishap (e.g. fire) and the severity of the mishap (killing the driver) is calculated. The risk level is determined by *merging* the probability of occurrence and the severity of the mishap. The "owner" of the system accepts or rejects the risk level. "Safety" is the level of acceptable risk (Haimes, 1998).

MIL-STD-882D defines "Safety" as: "Freedom from those conditions that can cause death, injury, occupational illness or damage to or loss of equipment or property, or damage to the environment (USN, 2000)."

As observed in above definition, system safety is concerned with those events /functions that can influence the software to be executed in such a manner that

would contribute to the loss of life, injury, loss of ship/missiles/assets, and equipment.

Mishap
Mishap is an unplanned and undesirable event or series of events resulting in death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment (USN, 2004).

Safety Critical
A term applied to a condition, event, operations, process, or item, whose proper recognition, control, sequencing, performance or tolerance is essential for safe system operation or use (USN, 2004).

Safety Kernel
An independent computer program that monitors the state of the system to determine when potentially unsafe system states may occur, or when transitions to potentially unsafe system states may occur. The safety kernel is designed to prevent the system from entering the unsafe state, and it is designed to return it to a known safe state (USN, 2004).

Risk Mitigation
Risk mitigation is defined as the process of avoiding, reducing and controlling, or deliberately accepting risk on the program (USN, 2004).

Software Safety
Hermann offers a detailed, practical definition:
"Features and procedures which ensure that a product performs predictably under normal and abnormal conditions, and the likelihood of an unplanned event occurring is minimized and its consequences controlled and contained, thereby preventing accidental injury or death, whether intentional or unintentional (Herrmann, 1999)."

System Safety

DoD believes, "System safety engineering is an integral element of systems engineering involving the application of scientific and engineering principles for the timely identification of hazards and initiation of the actions necessary to eliminate/control hazards or reduce the associated risk to an acceptable level" (DoD, 2002).

Stephans defines system safety as, "The discipline that uses systematic engineering and management techniques to aid in making systems safe throughout their life cycles" (Stephans, 2004).

The principal objective of a system safety program within the DoD is to ensure that safety consistent with mission requirements is designed into systems (USN, 2000), subsystems, equipment, and facilities. The approach to Systems Safety is

to design-out most of the potential accident causes while the system is still on the drawing board (Roland, 1990). The objective can be thought of as the balance between risk and controls.

Embedded in the above definitions, is system safety's role and responsibility with identification and mitigation of those events /functions that can influence the software to be executed in such a manner that would contribute to the loss of life, injury, loss of ship/missiles/assets, and equipment. The system safety effort is allocated for the system's life cycle, that is, development, production, operation, maintenance and disposal. The goal in system safety effort is to optimize safety, that is to minimize risk. This is usually achieved by early involvement, planning and designing safety into the system. This approach is in contrast with "reactive" approach that is studying of mishaps when they occur. System safety is, in practice, a function of program schedule and funding. The stakeholders are in control of these variables, and ultimately, in control of how safe is safe enough.

Failure Modes and Effects Analysis
A procedure by which each credible failure mode of each item from a low indenture level to the highest is analyzed using inductive logic (USN, 2004).

Residual Safety Risk
The remaining mishap risk that exists after all mitigation techniques have been implemented or exhausted, in accordance with the system safety design order of precedence.

Risk
A measure of inability to achieve program objectives within defined cost and schedule constraints (DoD, 2003). Risk has two components: the probability of occurrence of a mishap, and the consequence or severity of the occurrence of a mishap.

Causal Factor(s)
Causal factors are the result of poor or insufficient design, incorrect implementation of a good design, or potential or actual failures that would have to occur in order to result in the condition defined as a hazard. It is the probability of individual causal factors occurring and the logical relationship between those causal factors that determine the hazard likelihood. The logical relationship indicates whether the causal factors must occur together or may occur independently to lead to the hazard occurrence (DoD, 2003).

Hazard
Any real potential condition that can cause injury, illness, or death to personnel, damage to or loss of a system' equipment or property, or damage to the environment. A hazard is a prerequisite to a mishap (DoD, 2003).

Basic relationship between hazards, their causal factors and the top-level mishap is pictured in Figure 1.2.

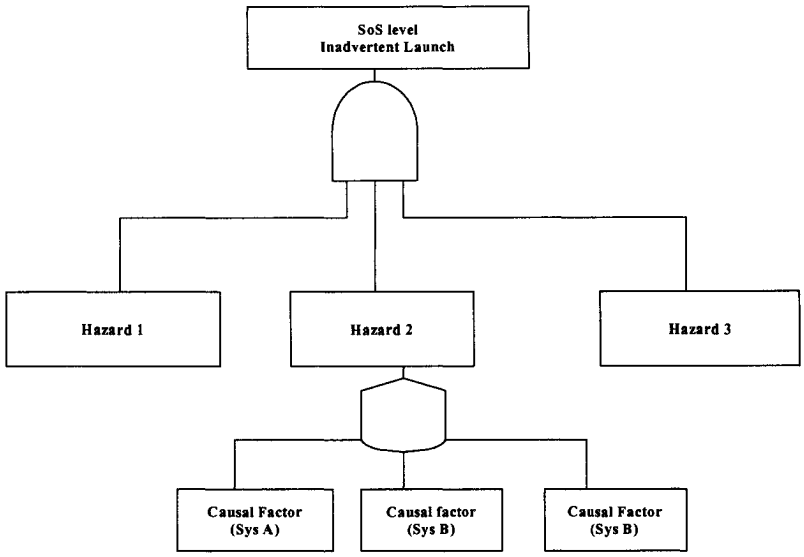

Figure 1.2 – Example of Fault Tree Analysis

Complex Systems/Systems of Systems

Keating, et al. (Keating C., Sousa-Posa, A., Mun, N., 2003) defines a System of Systems (SoS) as:

"A metasystem comprised of multiple embedded and interrelated autonomous complex subsystems that can be diverse in technology, context, operation, geography, and conceptual frame. These complex subsystems must function as an integrated metasystem to produce desirable results in performance to achieve a higher-level mission subject to constraints."

There are several terminologies being used to refer to a "system of systems". In DoD, terms like, "Combat System' or "Federation System" or "Family of systems' have been used. In private or academia, the term "meta-system" is used. All of the above terms are synonymous and refer to a larger complex system, made up by integration of a few single complex systems to achieve a new mission requirement.

The key word in the definition above is that the "sub-systems" are diverse. The diversity of these systems is not limited to operational nature or geography, but it expands to their computer languages, modeling of entities, technology, and environment in which they were produced to operate.

A "complex" system, e.g. a missile system, refers to a interrelated set of entities, functions, components that has decision-making capabilities, has many input and output, has high level of human interface, and interfaces with outside world. The "complexity" of the system increases when we increase the number of functional requirements, the interfaces, and human involvement. The increase in functional requirements and interfaces is made possible by more hardware, software, firmware, and by customizing needed operating environment. The increase in interfaces can include other systems, other services (Navy, Air Force, Army) or other militaries (allies). The more complex a system gets, the more "risk" is likely to have, and the more "need" for a system safety analysis exists. This direct correlation is shown in Figure 1.3.

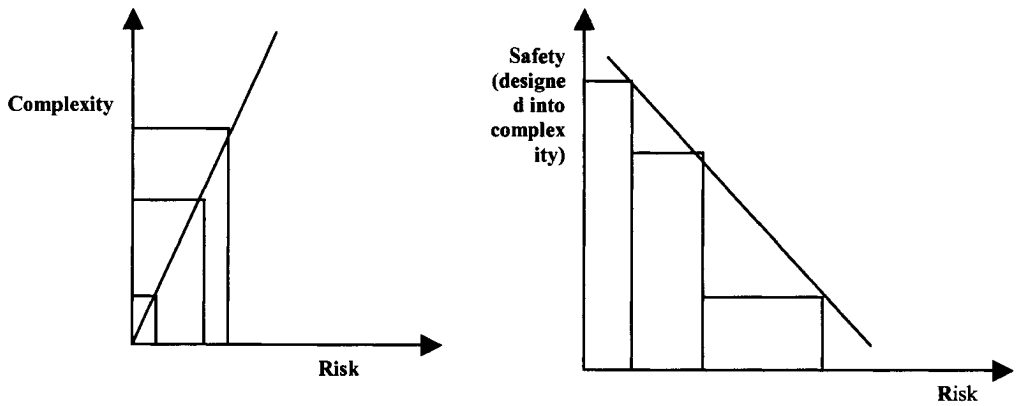

Figure 1.3- Complexity and Risk Relationship

The various complex systems are integrated and configured into one larger system to perform a new requirement. Figure 1.4 depicts this concept.

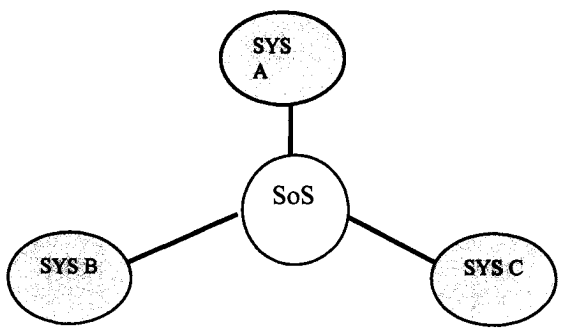

Figure 1.4 – Complex System of Systems

System of Systems Engineering

The integration and interrelation of multiple complex systems in multiple platforms, geography and/or within multiple policies, creates a new generation of complex systems problems. System of Systems Engineering (SoSE) is an attempt to address these new systems problems. It is an evolution of traditional systems engineering. SoSE is defined as (Keating, 2002):

"The design, deployment, operation, and transformation of higher level metasystems that must function as an integrated complex system to produce desirable results. These metasystems are themselves comprised of multiple autonomous embedded complex systems that can be diverse in technology, context, operation, geography, and conceptual frame."

Alborzi (Alborzi, 2004) illustrates the evolved systems engineering in Figure 1.5. This figure depicts the above definition where three single complex systems (A, B, and C) have been developed under separate Systems Engineering (SE) process and are now required to interoperate together as one larger system (SoS), thus are required to be modified, or transformed, and integrated via a systems of systems engineering (SoSE) process to perform a higher level of mission requirements.

System of Systems Safety Engineering

Currently, there exist a definition for System Safety Engineering.
MIL-STD-882 (USN, 2000) defines System Safety Engineering (SSE) as:

"The application of engineering and management principles, criteria, and techniques to achieve acceptable mishap risk, within the constraints of operational effectiveness and suitability, time, and cost, throughout all phases of the system life cycle."

As shown in Figure 1.5, SoS safety engineering will need to be an integrated part of overall SoSE.

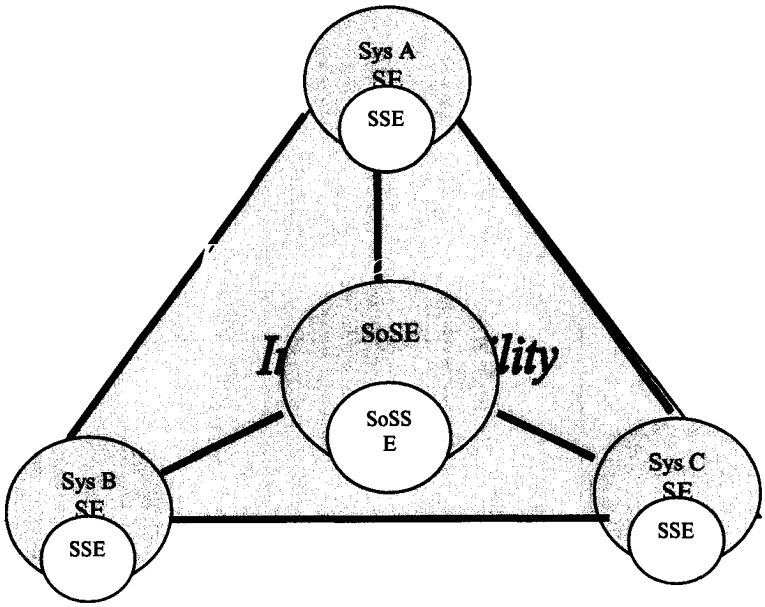

Figure 1.5- System of Systems Engineering (SoSE)

Interoperability

Interoperability has been defined by several entities from several engineering or science spectrum. Here are a few:

"The interoperability is the capability of N heterogeneous systems to conform to a set of ISO standards of the same profile and to communicate according to these standards in an environment representative of the reality"(Batard, 1991).

Wileden and Kaplan define interoperability from software perspective, as, *"the ability for multiple software components written in different programming languages (e.g. C++, JAVA, etc.,) to communicate and interact with another"* (Wileden, 1997).

Alborzi defines the interoperability, from weapons systems perspective, as the, *"Integration and interoperation of multiple new and legacy systems for new higher level mission requirements"* (Alborzi, 2004).

In general, interoperability is defined as, "To exchange information and perform joint tasks (Young, 2002)."

The understanding of the notion of interoperability can be facilitated by Figure 1.6 below. AEGIS Combat System in joint cooperative tasking with Ground-based Midcourse Defense program (GMD), linked with secure satellite communication systems performing C4I tactical operations.

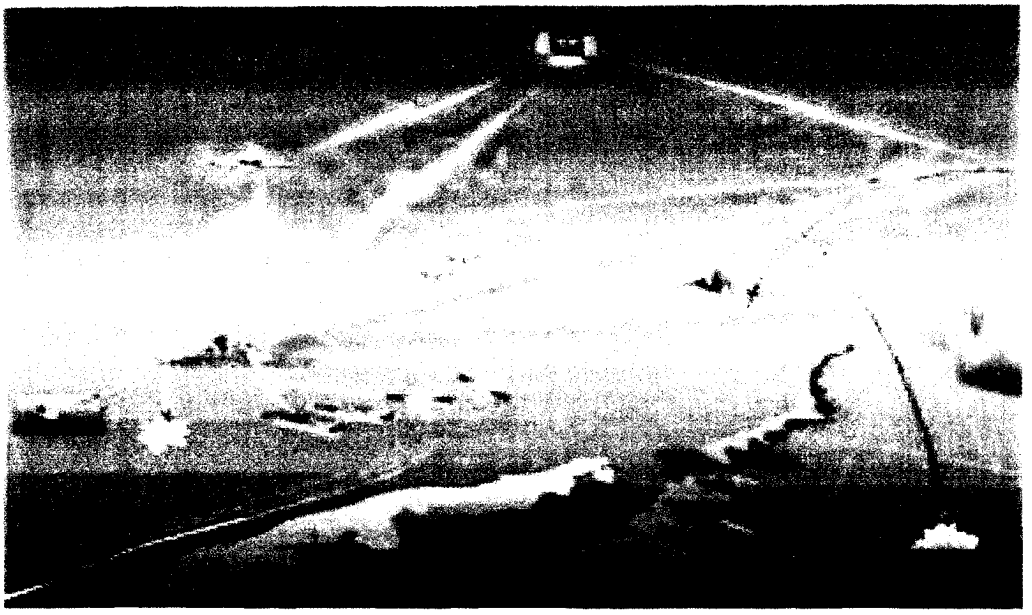

Figure 1.6- Interoperability Conceptual View **(Aegis, 2004)**

Safety Interoperability

Imagine a scenario where two similar systems are interconnected to pass target data for joint target engagement. Due to interface limitations and/or differences in data requirements, the two systems cannot communicate effectively. To remedy this "incomparability" and making these systems work together, DoD contracts a developer to build a "translator medium"-- a communication system that can be added to the configuration whose function is to "translate" the target data of one system and present it to other system in the language or the format it understands. Since this intermediate system is only a communication system (no connection or interaction to weapons systems), it is believed to not have any safety critical functions, therefore it is not considered for system safety study, and the software development continues till completed and interconnected with the other two systems. Now the two systems work together well and can pass target data. However, under a certain and often unknown situation, a software "glitch" in the intermediate system, causes it to use the position coordinates of one system to be passed as target coordinates to the other system. The result can be catastrophic if engagement and the release of a weapon would proceed.

The term, "safety interoperability" is a new term developed for the purpose of this research. It intends to communicate the concept of interoperability from *safety* perspective. As such, there are no published formal definitions of "safety interoperability"; however, the author has developed the following definition:

"A capability encompassing many of the safety issues relative to integration, compatibility and interface that are impinging upon the effectiveness with which independent, heterogeneous and/or homogeneous systems, components or elements, including human factor, may safely interact"(Alborzi, 2004).

1.7 OVERVIEW OF THE SYSTEM SAFETY INTEROPERABILITY FRAMEWORK (SSIF)

To facilitate the understanding and the usability of SSIF, we must first illustrate the general architecture of a complex combat system, starting first from single system view.

System A is a complex weapon system, e.g. a radar system, which is procured and developed using the traditional systems engineering (SE) processes. This means, the system will have a Concept of Operations (CONOPS), system requirements, design and production, and finally installation and maintenance. As a integrated part of systems engineering Integrated Product Team (IPT), as shown in Figure 1.8, system safety engineering (SSE) will be conducting its own system-based engineering evaluation using the SE IPT interfaces (see Figure 1.7) and artifacts for the purpose of identification, elimination or control of system's hazards. This evaluation is conducted near parallel to the program development. The evaluation includes requirements analyses, design and testing evaluations.

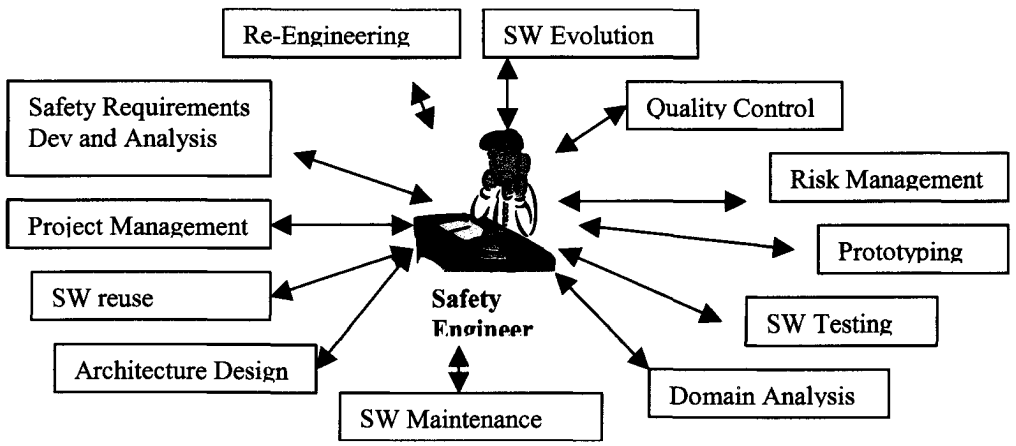

Figure 1.7- System Safety Engineering (SSE) Interactions with SE

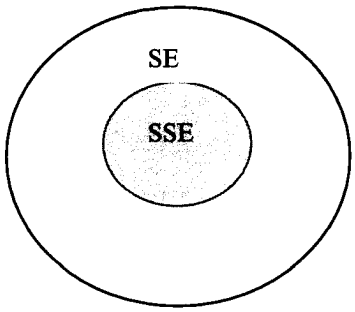

Figure 1.8- Integration of SSE into System Engineering (SE)

System B is another complex system, perhaps a command and control (C2) system. This system may be legacy or new, may be in the same platform as System A or not. This system has most likely been through a complete SE and SSE also.

System C is yet another complex system, perhaps a missile system, used in the same platform or other than system A or B, legacy or new, and it is assumed that has been evaluated under its own SE and SSE.

Figure 1.9 demonstrates the complex systems which have been developed independently, may have different environment in which they can operate, may have different computer languages, different output requirements and may have used different technology, tools, etc.

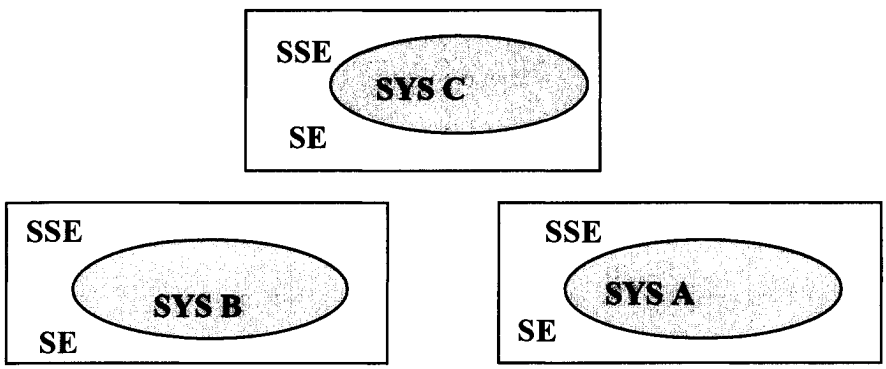

Figure 1.9- Diverse Complex Systems

New Architecture- A System of Systems-

Systems A, B and C are configured and integrated to represent an overarching system- a system made up of few complex systems working and interoperating

together as a larger more capable system. As an example of operational concept (CONOPS), system A may detect a target that seems to fit a threat profile, using communication an intelligent systems, this information is passed on to System B. System B evaluates the threat, conducts an identification and sends an engagement order to System C for prosecution.

This integration and interoperation of various complex systems working as one larger, more complex meta-system is made possible by a SoS engineering methodology. This methodology, similar to the single-system SE mentioned above, will employ the design and development teams, but it will have a different focus, attempting to solve different system problems and striving for different operational outcome.

On the same token, system safety engineering team will need to focus on system of systems architecture, and attempts to evaluate the hazards in the overall meta-system. This will mean the safety engineering methodology used during the development of single systems is now inadequate in this new, larger, more complex, and more fluid context. SoS safety engineering (SoSSE) will need to be integrated and embedded into the SoSE. The underlying assumption is that the interoperability impediments have been resolved in SoSE (perhaps as the part of SoSE methodology). The example mechanisms by which this resolution is possible are Object-Oriented Method for Interoperability (OOMI) (Young, 2002) and Holistic Framework for Software Engineering (HFSE) (Puett, 2003).

Figure 1.10 shows the integration of SoSSE into overarching system engineering process- SoSE for this new architecture. The new safety engineering interactions is demonstrated in Figure 1.11.

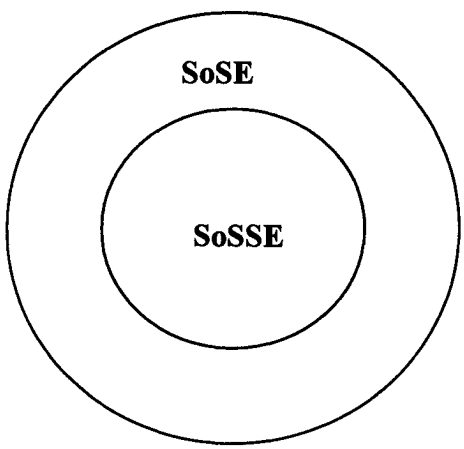

Figure 1- 10 - Integrated SoS Safety Engineering

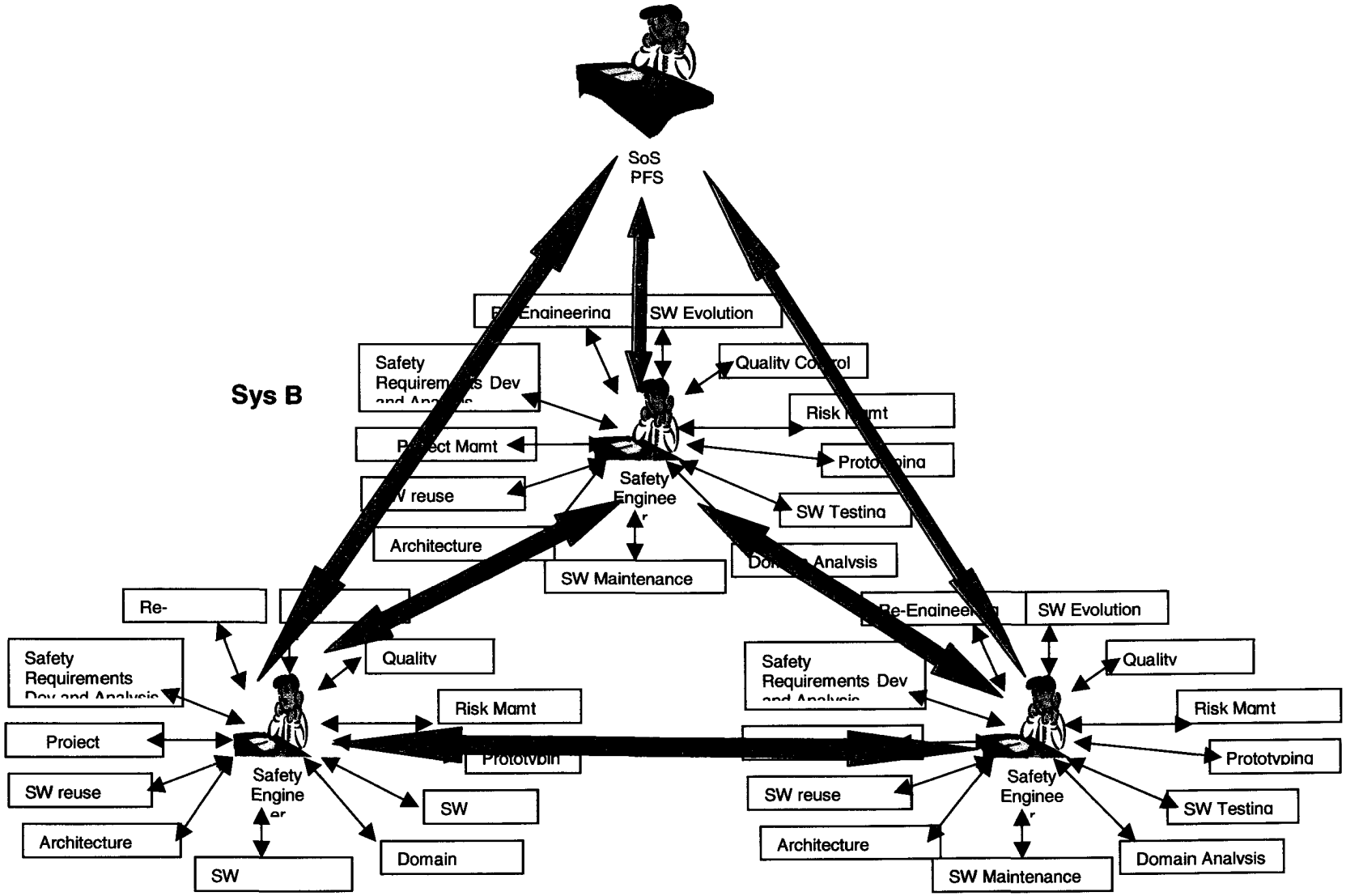

Figure 1.11 – Safety Engineering Interactions in SoS Architecture

System Safety Interoperability Framework

A new framework is needed to ensure the application of engineering techniques and analyses to the overall Navy's SoS to identify and eliminate hazards that impede the proper interoperation of safety critical data (SCD) or functions. Safety critical functions are those whose improper operations (or failure of operations) may contribute or lead to a mishap. This framework will integrate SoS Safety Engineering (SoSSE) into the SoS Engineering (SoSE) discipline, so that we can ensure the implementation of SoS requirements with safety impact, and design in the safety nets necessary to preclude the unsafe operation of SCD during the interaction of various diverse complex systems. The System Safety Interoperability Framework (SSIF) will provide us this vision, and it is illustrated in Figure 1.12.

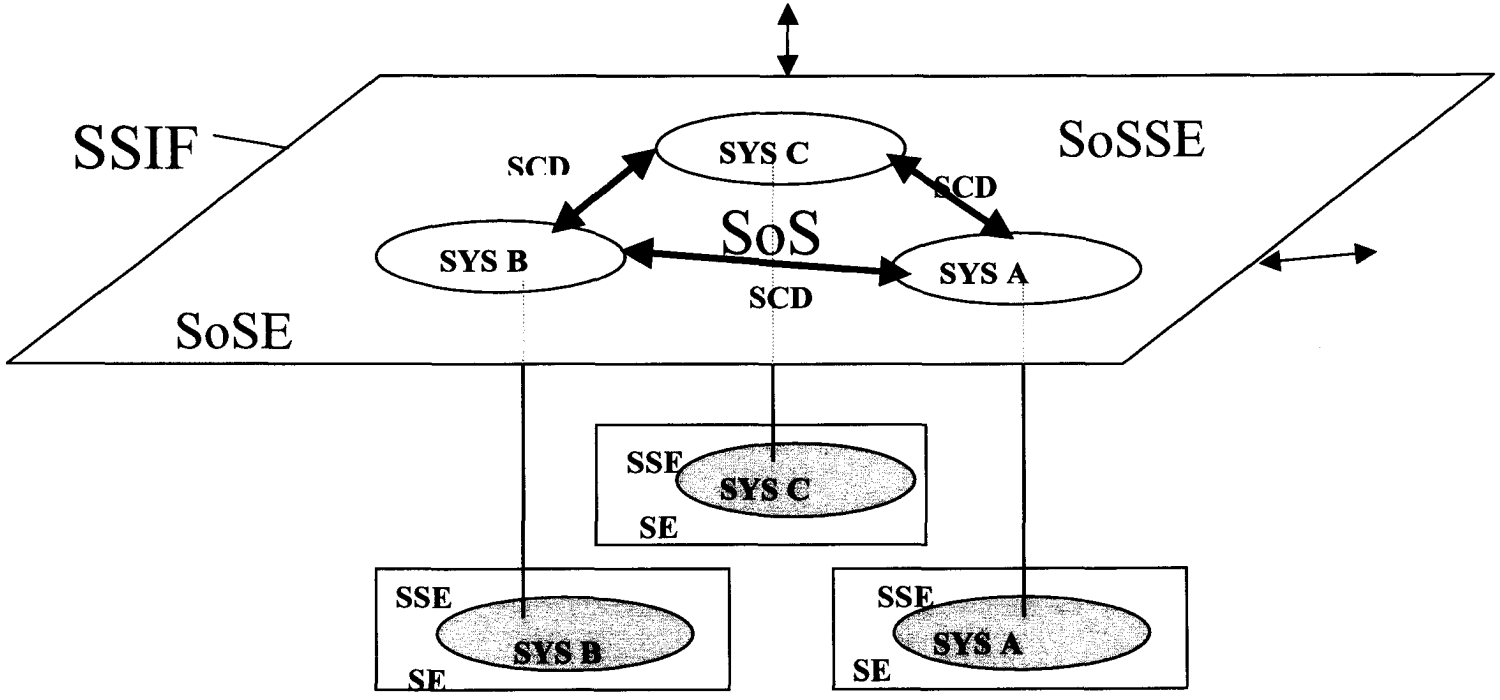

Figure 1.12- System Safety Interoperability Framework (SSIF)

1.8 RESEARCH CONTRIBUTIONS AND SIGNIFICANCE

1.8.1 Accomplishment of Research Objectives

The review of this dissertation will reveal that the research objectives listed in paragraph 1.3.1 have been accomplished.

1.8.2 Contributions

The analysis of system safety interoperability has made the following contributions to the "Practice" segment of Body of Knowledge:

a. Identification and evaluation of the safety issues in external interfaces and external interactions provides a basis for focused and orchestrated safety analysis from total system perspective.

b. The analysis of the focused area in project management team will translate into allocating the right resources (both from expertise and funding perspective); this means doing the necessary work with optimum efficiency.

c. Identification of the safety attributes of the system that are affected by the integration into a system of systems. The safety attributes are those features, design constraints, etc. that define the safety in system context.

d. The "building blocks" created by this research and the analytical methods used will provide more insight into the safety interoperability issues than has traditional methods.

e. The System Safety Interoperability Framework (SSIF) can be used for mishaps risk reduction purposes.

f. Risk reduction noted in "e" above will translate into saving lives of our war-fighters by reducing and eventually eliminating friendly fire accidents.

The contributions to the "Theory" segment of Body of Knowledge include:

a. Defining and documenting, for the safety interoperability impediments in a combat system tactical environment. This will aid the future design engineers in designing safety during development of complex systems.
b. Posit a statement of system safety interoperability for the Navy's combat systems.

c. The advancement of "System Thinking" by proving, through scientific analyses, that interoperability capability is not just a performance issue but also a safety issue that needs to be included in all systemic evaluations. The new "improved" systems thinking will enable us to identify processes that have potential in solving complex problems, will also enable us to have control over the "health" of the system.

1.8.3 Significance

The significance of this analysis lies in its somewhat 'invisible' yet crucial role in development of weapons systems and their use in tactical operations. Primarily, the following lists the significance of study:

a. Originality of Concept- Interoperability is not just a Performance issue, but also a safety issue. Although DoD have initiated the effort in understanding and implementing interoperability requirements in today's complex systems, the system safety dimension of it has not been studied or even acknowledged.

b. A new leap into system safety engineering. Up to this point, system safety engineering had a single-system-based approach. No provisions, tools or processes or standards exist for combat system of systems architecture. No scholarly analysis of system safety interoperability in the Navy's systems of systems tactical environment has ever been conducted.

c. Risk Reduction/Management- The analysis with this depth and breath will have significant impact on how engineers design safety into the systems, so safety can become the property of the meta-system. Safe systems have inherently low risk for mishaps. This will translate into lower number of friendly fires, inadvertent launch, or launch based on misinformation, miscommunication, etc.

d. System safety being advanced in "systems thinking". The advanced systems thinking will lead to the development of systems theory, application methods and finally in higher systems thinking.

1.9 DISSERTATION ORGANIZATION

This dissertation is organized in a way to facilitate the reader's understanding of concepts through the development of the SSIF and through the analysis of the use case, Ballistic missile Defense (BMD) 3.0 program.

Chapter I introduces the readers to the concept of interoperability and system safety, and why we need a well-undertaken scholarly research on the subject. It points out the complexity of the new architecture- namely systems of systems and

the need for a more structured, safe and reliable way of procuring and using weapon systems.

This chapter also presents a brief overview of the SSIF, and it provides the definitions of terms and concepts, the purpose of study, the objectives and the contributions made by this research.

Chapter II presents a survey of current literature in the field. The literature review is divided into three parts:

a) System of systems
b) System safety
c) Interoperability

As a major part of this section, the "gap" in literature is determined and presented. Previous advancements in the area of interoperability such as Holistic Framework for Software Engineering (HFSE) and Object-Oriented Model for Interoperability (OOMI) is discussed and used as foundation in this research.

Additionally, in Chapter II, the system safety interoperability issues are identified as the result of literature review.

Chapter III presents the System Safety Interoperability Framework (SSIF) in detail. Included in this presentation are the SSIF characterization attributes and what each attribute entails.

Chapter IV presents the analysis conducted for AEGIS BMD 3.0 program using SSIF. As a part of analysis, the chosen "abbreviated" methodology used for the analysis will be discussed. The "findings" will be stated as well.

The results are also provided in this chapter. This chapter also includes the system safety interoperability statement that is "emerged" as the result of analysis. This statement will be a significant contribution to the engineering community, in particular, to the safety community.

Chapter V covers the conclusion and recommendations. In this chapter an overview of the research is presented. The purpose and the research questions are reviewed. It is evidenced that the objectives of the research, research questions and the purpose were satisfied. Moreover, the contributions and the significance of the research are presented, and recommendations for future scholarly study and research are provided.

Appendix A provides the official authorization from U.S. Navy to use the BMD program for this analysis and to publish this dissertation.

Appendix B contains the written approval and validation of SSIF by three subject matter experts. This appendix includes the processes, the criteria used, and a brief biography of each expert.

CHAPTER II

REVIEW OF LITERATURE

"Its mistakes of commission have been legion; and its mistakes of omission have been even greater. It has all too often done nothing when it should have realized that problems cannot be avoided by refusing to admit they exist."

Harry S. Truman
U.S. Senate Special Committee to
Investigate the National Defense program
January 1942

2.1 COMPLEX SYSTEMS OF SYSTEMS

The engineering of complex systems of systems has received special importance in recent years. Systems of systems or combat systems terminology is now widely used to describe the combined operation of many platforms, weapon systems, and communication systems to achieve an overall joint tactical objective in military operations. The increased complexity is felt at all levels of command.

General Sheehan, former Commander in Chief of U.S. Atlantic Forces, in 1997, said,

"Victory will depend on the ability to master the 'system of systems' composed of multi-service hard-and sot-kill capabilities linked by advanced information technologies."

The concept of systems of systems came about as a cost '*mitigator*'. How can we use the synergism available by combining the strengths of each system towards a broader mission objective? The answer is system of systems integration.

2.1.1 System of Systems Engineering

To create a new system of systems, we need to have a system of systems engineering.
Keating, et.al., states that System of Systems Engineering (SoSE) is in its embryonic stages of development (Keating, Sousa-Posa, and Mun, 2003). Traditional systems engineering approaches have been successful in addressing the complex system problems with technical solutions (Sage, 2000).

However, there have been new complex problems emerging that do not have single technical solutions, but a "satisficing" outcome (Alborzi, 2004). Keating, et. al. attempt to establish a foundation for SoSE methodology on which these new system problems can be addressed and resolved (Keating, Sousa-Posa, and

Mun, 2003). Table 2.1 (Keating, Sousa-Posa, and Mun, 2003) shows the progression of development of systems of systems engineering.

TABLE 2.1- Development of Systems of Systems Engineering Capability

Level	Development Area	System(s) of Systems Engineering Development
1	Supporting Systems Engineering Processes, Tools, and Methods	The systematic approaches and supporting tools that facilitated the accomplishment of systems engineering by individuals units, and organizations.
2	SoSE Principles	The fundamental laws that govern, define, explain, and predict the structure, behavior or performance of systems of systems.
3	SoSE Local Applications	The performance of system of systems engineering for a specific problems or issue in a localized setting.
4	SoSE Methodology	A generalized and theoretically grounded framework that guides the orderly design, assessment, or transformation of a system of systems.
5	Systemic SoSE Philosophy	A fundamental worldview based in systems principles that drive decision, action, and interpretation in all aspects of system of systems engineering.
6	SoSE Culture	The artifacts, values and beliefs, and fundamental assumptions that permeate the accomplishment of SoSE in a unit or organization.
7	SoSE Research	Engagement in the rigorous study of SoSE to generate, acquire, interpret, and disseminate knowledge to enhance the performance of SoSE.

In addition, Sousa Posa and Keating (Keating, Sousa-Posa, and Mun, 2003) provide guiding system principles for SoSE methodology development. These are Systems Principles that need to be in consideration for basis of SoSE methodology development. To clarify this idea, these guiding principles are intended to stimulate thinking, enhance understanding, and provide an insight into the foundations for the SoSE methodology (Keating, Sousa-Posa, and Mun, 2003). These are:

- Compatibility
- Minimum Critical Specification

- System Control
- System Context
- Boundary Establishment
- System Outcome Achievement
- Complex Systems transformations
- Iteration
- Unity of System Purpose
- Self-Organization
- System Viability
- Complementarity

SoSE Methodology

A SoSE methodology provides a basis from which a new system Federation can be designed and deployed, or an existing Federation can be transformed (Keating, Sousa-Posa, and Mun, 2003). This methodology is intended to be flexible and not prescriptive as traditional system engineering methods (Gibson, 1991), and it is intended to be able to adjust to new emergent problems unique to SoSE.

Keating and Sousa-Posa have attempted to lay the groundwork for a SoSE methodology that is clear and guiding, yet not prescriptive, general and high-level, yet with sufficient details that would enable the engineers to guide the resolution of a complex systems engineering problem, and moves the engineers forward, yet not in linear manner, but in iterative, spiral manner. Figure 2.1 depicts the initial attempt in developing a SoSE methodology by Keating and Sousa-Posa (Keating, Sousa-Posa, and Mun, 2003).

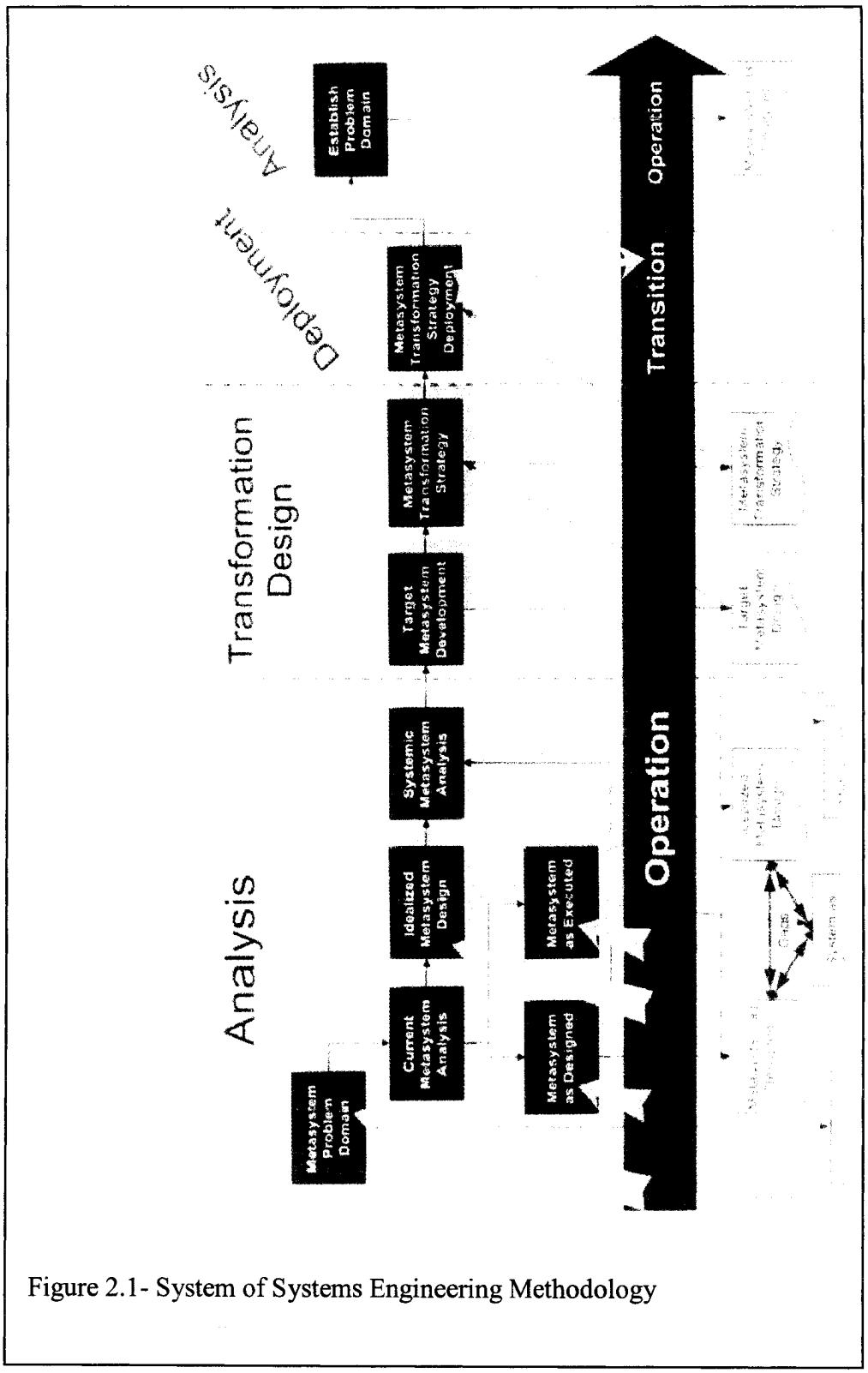

Figure 2.1- System of Systems Engineering Methodology

2.2 INTEROPERABILITY

The interoperability is defined as:

The ability of systems, units or forces to provide services to and accept services from other systems, units or forces and to use the services so exchanged to enable them to operate effectively together (Pridmore, 2000).

Col. Marek Amanowicz, from Military University of Technology in Poland, provides one of the few interoperability models available. He focuses on one of the many dimensions of interoperability, and that is the Communications and Information Systems.

Figure 2.2 (Amanowicz, 1996) shows the concept of interoperability modeling.

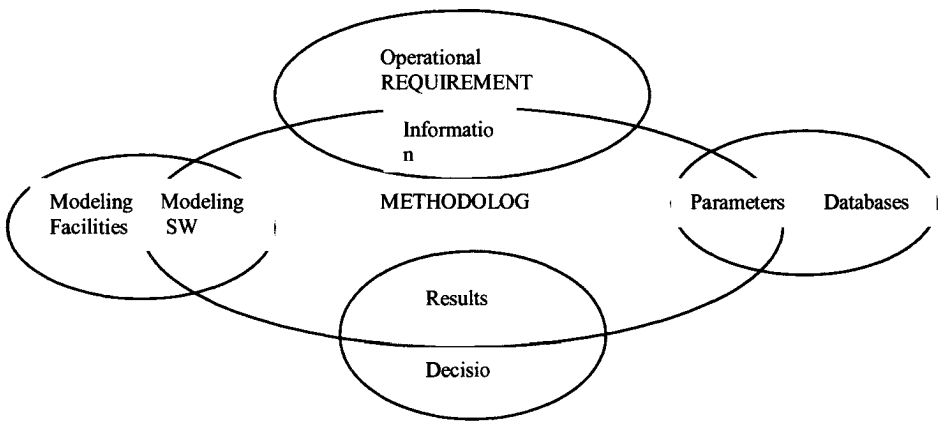

Figure 2.2- Concept of Interoperability Modeling

U.S. Navy's legacy systems were developed in centralized (Aiken, 1994), terminal to host architecture (Chia-Chu, 2001). Typically, the entire system functionality was placed on the mainframes. Chia-chu (Chia-Chu, 2001) illustrates this in Figure 2.3.

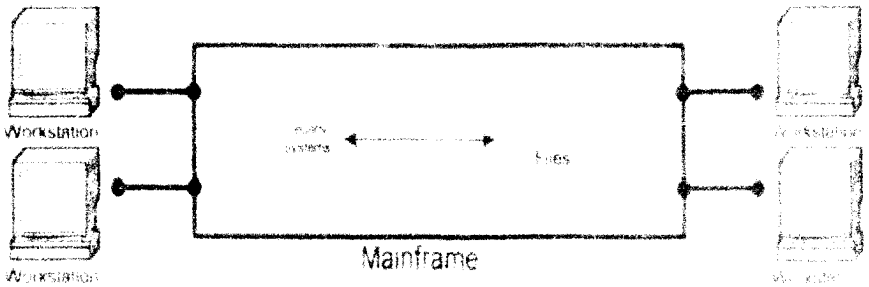

Figure 2.3- A Centralized Computing Environment for Legacy Systems

For U.S. Navy's AEGIS legacy baselines these mainframe were UYK-7s and UYK-40s. Application processing logic, the data resources and displays/presentations were all hosted on the mainframe. The application was tightly integrated in millions lines of source code rather than in smaller, modules. Changing legacy systems software is a nightmare, because thousand of lines may be affected by changing a 10-line function. In recent years, however, U.S. Navy has begun transforming its systems to a more distributed, object-based architecture. The first of these initiatives were conversion of legacy systems, written in Navy exclusive language, CMS-2 to Ada. The Commercial-Off-The - Shelves (COTS) hardware replaces the old UYKs. And recently, the Navy is moving toward converting its system to Open Architecture, using Object Oriented Architectural Design (OOAD). Middleware such as Common Object Request Broker Architecture (CORBA), Component Object Model/ Distributed Component Object Model (COM/DCOM) and Java RMI make the migration of the legacy system to a heterogeneous distributed computing environment possible (Chia-Chu, 2001).

AEGIS Combat System approach in distributing its computing applications includes using Java.

With requirement of interoperability in forefront of U.S. Navy war-fighting strategy, the transformation and use of new and legacy systems in a different environment, under totally different contextual requirements is inevitable.

Various aspects of interoperability requirement have been researched. For example, the development of a large-scale interoperable database system, operating in a dynamic environment has been addressed by many researchers. This interoperable database system should provide uniform access to heterogeneous information sources (Ling, 1997). Li and McLeod write on the importance of management of the inherent interdependencies among data in different databases, in supporting information sharing in federated database systems (McLeod, Li, 1993). They note that data in such databases can implicitly and/or explicitly exhibit various forms of interdependence, such as existence,

structural, and functional/ behavioral dependencies. The ability to capture and systematically support, such intrinsic interdependency relationship is essential systems (McLeod, Li, 1993).

There seem to be two solutions for the problem of interoperability—integrated systems and open systems. This thought leads us to our next discussion, the Object-Oriented Model for Interoperability that will show the balance between 'opening' the systems and integrating them to allow seamless interoperation to occur between legacy, new or hybrid systems. However, before that, it is necessary to review the root causes behind un-interoperability, that is the causes of heterogeneities and what are the consequences of integrating the heterogeneous systems that were developed independently in isolation and with no interoperability requirement in the first place. Paragraphs 2.2.1 and 2.2.2 will address these issues and will offer a method for resolution.

2.2.1 Causes of un-interoperability

In Figure 2.4, Young (Young, 2002) provides an example of how war-fighting systems may be interconnected to provide a larger system with an increased capability than any of subsystems.

As shown in the figure, the forward observer's Battlefield Digital Assistant (BDA), implemented on a Palm wireless hand-held device, represents the target using a *MechanizedCombatVehicle* record structure with elements *mcvType* used to indicate whether the vehicle is a tank, personnel carrier, or reconnaissance vehicle; *mcvLocation* provides the vehicle's location using Military Grid Reference System (MGRS) coordinates; *mcvTime* giving the time of observation at the specified location in Local Mean Time (LMT); and *mcvRadius* specifying the maneuvering range of the vehicle in kilometers (km).

The Command and Control, Computers, Communication and Intelligence (C4I) system within the federation, represent the same target using an *ArmoredMilitaryVehicle* structure containing elements *designation* specifying the type of the vehicle (main battle tank, missile launcher, armored personnel carrier), *position* element that provided the vehicle's coordinated using latitude and longitude; and *time* in Greenwich Mean Time (GMT) providing the moment when the vehicle was observed. Its application is most likely written in C++.

The third system in the federation, e.g. Tomahawk Planning System, written in Ada, and uses *ArmoredFightingVehicle* record structure to model the targeted tank. The record includes *afvClassification* specifying the type of vehicle (battle tank, rocket launcher, truck), *afvLocation* providing the vehicle coordinates using latitude and longitude, *afvObsTime* giving the time of observation of the vehicle at the specified location in GMT, and *afcStatus* indicating whether the vehicle is operational, damaged or destroyed.

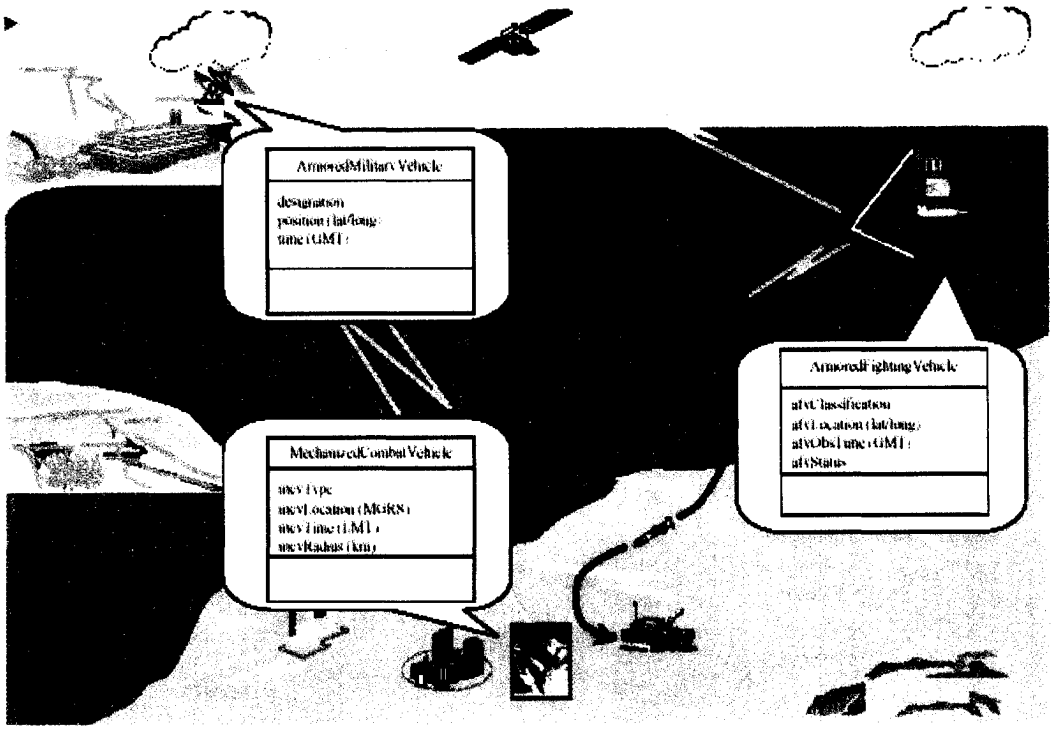

Figure 2.4- Impediments to Systems Interoperability (Young, 2002)

Batinin offers the following root causes for modeling differences.

Causes of Data Modeling Differences
Batinin, et. al., (Batinin, Lenzerini, 1986) describes three different causes for representational heterogeneity:

Different perspective. The different needs of users, program managers, and design teams can lead to differences in data representation even when modeling the same information.

Equivalent constructs. Equivalent models of the same real-world domain can be created using different combination of the same basic modeling constructs.

Incompatible design specification. Different application design specifications can result in different database schemas for the same real-world domain.

Although Batinin cited these causes in the context of database schema integration, they also apply to the types of model heterogeneity that is evident in complex systems.

Kinds of Data Modeling Differences

Several authors have noted the modeling differences in databases used in heterogeneous systems. Wiederhold (Wiederhold, 1993) developed seven classes of heterogeneity. Although these were defined for database systems, they also apply to complex systems of systems with interoperability requirement. Young (Young, 2002) adds a new kind- the "heterogeneity of structure" to the list. Therefore, the classifications of modeling differences include:

- Heterogeneity of Hardware and Operating Systems
- Heterogeneity of Organizational Models
- Heterogeneity of Structure
- Heterogeneity of Presentation
- Heterogeneity of Meaning
- Heterogeneity of Scope
- Heterogeneity of Level of Abstraction
- Heterogeneity of Temporal Validity

Heterogeneity of Hardware and Operating Systems

These are differences in operating systems and the hardware used when new, legacy or hybrid complex systems are integrated. The speed in technology advances makes this problem a certain one in a federation of systems. The hardware platform differences include how the same information is displayed in different systems, such as the size, length, and format of the data. The differences in operating systems have been a more "familiar" problem. In DoD, the two most utilized operating systems in the world of COTS are Microsoft Window and UNIX products.

Heterogeneity of Organizational Models

The organizational modeling difference refers to differences in the conceptual modes used by independently developed complex systems. In the context of database systems, the organizational modeling differences refer to differences in database model such as hierarchical, relational, universal or object structured (Hammer, 1999). In interoperability, it refers to differences in analysis and design principles used such as Object-Oriented Analysis and Design (OOAD) or structured analysis approach (Pressman, Young, 2001).

Heterogeneity of Structure

This modeling difference refers to differences in structural composition, schema, or implied vs. explicitly stated information. Young (Young, 2002) explains that this difference can arise when a real-world entity is modeled as an object on one system and as attribute in another, e.g. such as an aircraft route being modeled as an attribute of an aircraft mission object on one system (as one element of the

overall mission) and as a separate entity used for de-conflicting missions in another (Holowczak, 1996). It will be shown later that "de-confliction" is a safety critical issue.
The heterogeneity of structure also refers to concepts being modeled differently in the schemas of corresponding systems, e.g. a relationship that is modeled as one to one in one schema and one-to many in another (Hammer, 1999). This type of mismatch is especially apparent in Identification friend or Foe (IFF) interrogation tasks. IFF processing is noted later in this section as a safety related issue.

Heterogeneity of Presentation
This heterogeneity is related to domain mismatch problems such as different units of measurement, difference in precision, disparate data types, and different field lengths or issues of constraint integrity. An example is when one system measures the position in latitude and longitude, and another system measures the position with Military grid Reference System (MGRS) (Hammer, 1999). Systems may also use different units of measure when quantifying the same object such as yards in one system and meter in another. Disparate data types refer to when a system represents a phone number as an integer and another system uses alphabet (a string of characters) for the same object. Systems may use different length for their words (Wiederhold, 1993), for example one system may provide a 20 character description of a function, the other may choose to use only ten characters. Different systems may employ different constraints on the same parameter, for example one system may choose a 50 miles range away from the own-ship to be the safe zone, and another may choose a 30 mile range for the same parameter.

Heterogeneity of Meaning
This heterogeneity relates to problems with spoken language, the semantics and syntaxes. Homonyms is using the same word to convey different meanings, and synonyms are different words that have the same meaning. The use of abbreviations by different systems is included in this heterogeneity, for example, a system may use "POS" for position, and another system may use "PSIT" to refer to the position of an object.

Heterogeneity of Scope
This heterogeneity refers to the scope of information being used by different systems for the same entity. These differences can arise from different perspectives on what needs to be captured by a given application about the real-world object (Wiederhold, 1993, Holowczak, 1996).
For example, a logistics management system (Young, 2002) may use attributes like *fuelCapacity* and *ammunitionStatus* for a battle tank, and a Command and Control (C2) system is likely to use *weaponRange* and *defensiveArmor* in its tank model.

Heterogeneity of Level of Abstraction
This heterogeneity refers to differences in the level of abstraction given to an object being modeled. For example, one system may sum profits earned in a monthly total, and another system may aggregates the same basic idea in yearly total (Wiederhold, 1993).

Heterogeneity of Temporal Validity
The modeling difference arises from differences in the time used by two models to observe or record the state of a real-world entity (Young, 2002). These temporal validity issues are particularly an issue with military C4I systems (Holowczak, 1996, Wiederhold, 1993). For example in one C4I system, the satellite picture of a threat area may be kept for a month to be valid (and be used), and the same picture in another system may kept for a year to be valid (and used). Validity of this real-world entity may have further implications depending on how it is used.

2.2.2 Object-Oriented Method for Interoperability

Paul Young provides a method for resolving the causes of un-interoperability. In Paragraph 2.2.1, the various kinds of heterogeneity in legacy systems are explained in detail. These heterogeneities cause the systems to be unable to interoperate effectively with each other. Young elected an approach that included aspects from formulae, ontology, and model methodologies to provide the optimum benefit in resolving heterogeneities among a federation of independently developed systems (Young, 2002). This approach is called Object-Oriented Method for Interoperability (OOMI) and is based on a model of the real-world entities whose state and behavior are shared among systems. The OOMI will be used in this research to serve as foundation on which System Safety Interoperability Framework will stand, and so it will be discussed here in great detail.

The OOMI methodology includes Federation Ontology to provide a canonical representation for the shared information, and utilizes formulae-based methods for resolving differences between components and canonical representations of the shared information (Young, 2003). This method takes advantage of the following capabilities:

- Object-Oriented Analysis and Design (OOAD)
- Federation Interoperability Object Model (FIOM)
- Integrated Development Environment (IDE)
- Translators

OOAD - provides principles of abstraction, information hiding, and inheritance that can be employed in the resolution of differences among independently developed systems (Khoshafian, Abnous, 1995, Walsh, Couch, 2000).

FIOM - provides an abstract model of the real-world entities whose state and behavior is shared among federation systems, hiding the details of how that information is modeled on different systems, except when required for difference resolution. It is constructed prior to runtime. FIOM provides the different component system models of the shared real-world entities with their abstract model in order to facilitate resolutions. It also provides the means for resolving modeling differences among systems, and it is intended to be extensible, that is, adding new entities to the federation or including new component system models of a real-world entity, shall not affect contents or relationships in an existing model.

IDE - Construction of a FIOM for a large system of systems can be time-consuming task as well as prone to error. An Integrated Development Environment (IDE) can automate this process (Young, 2002).

Translators- since the resolution of heterogeneity needs to be resolved during runtime, and FIOM is constructed prior to the runtime, there is a need for translators that can act as intermediaries between federation systems and translate the FIOM input during runtime.

Figure 2.5 provides an outline of OOMI (Young, 2002).

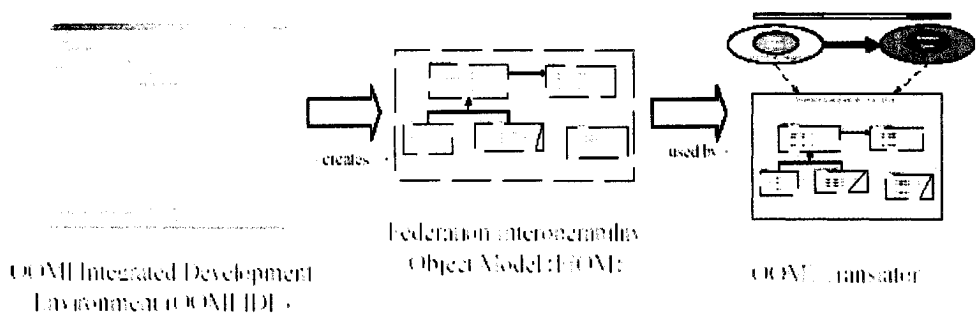

Figure 2.5- Object-Oriented Method for Interoperability (OOMI)

The key components are described in the following section in more detail.

OOMI Components:

IDE

In order to resolve heterogeneities among systems, a model of the real-world entities involved in the interoperation, is constructed prior to the runtime. This model is FIOM as stated earlier. IDE is used to assist the construction of FIOM. IDE is GUI-based and it is used to (Young, 2002):

- Identify the real-world entities involved in the interoperation of systems in a federation.
- Specify the different views of a real-world entity resulting from dissimilar component system perspectives of the attributes and operations required to model an entity.
- Define a "standard" federation representation for each real-world entity view identified and establish the relationship between the "standard" view representation and various subsystem view representations.
- Construct an inheritance hierarchy relating the different views of a real world entity.
- Manage ontology of terms and representations for the Federation to be used for defining the "standard" federation representation of the real-world entities whose state and behavior are shared among federation systems.
- Define the transformation required to translate between sub-system and "standard" representations of a view.
- Generate system-specific information to be used by a translator to resolve modeling differences between sub-systems during their runtime operations.

IDE allows the engineer to extract, from external interfaces, the entities that will be involved in interoperation. Each federation entity (FE) construction is assisted by Federation Ontology containing the accepted terminology and representation to be used for defining the federation model of the real-world entity specified by an FE. Moreover, the IDE provides functionality for accessing and modifying the Federation Ontology during FIOM construction (Young, 2002).

Additionally, correlation software is used during the FIOM construction to establish relationships between different views of a FE. This will define the inheritance hierarchy that will be used by FIOM.

After different representation, views of an FE is identified, the transformations required to translate between representations must be defined. IDE uses a GUI-based matching process (Young, 2002) to provide automated transformation development, and the maintenance of the translation library for future use.

The class transformation and relationship information is extracted from FIOM for each subsystem.

NOTE: Detailed description of IDE functions can be accessed by obtaining the cited reference.

FIOM

To understand and appreciate the capability of FIOM in constructing the real-world entity of system federation, it is necessary to review paragraph 2.2.1- i.e., the kinds of modeling differences that can occur in heterogeneous systems.

The heterogeneities of independently developed systems included (Young, 2002):

- Hardware and operating Systems
- Organizational Models
- Structure
- Presentation
- Meaning
- Scope
- Level of Abstraction
- Temporal Validity

When two systems use different features to model the same real-world entity, then two systems are said to have different *views* of that entity. When two systems use the same features to model a real-world entity, it can be said that the two systems have the same view of the entity. The two systems that have the same view of a real-world entity can model that entity in different ways, that is, the two systems have different *representations.*

A view is defined in OOMI as tuple ($A\epsilon$, $\Omega\epsilon$) (Young, 2002) of attribute and operation sets used to model the state and behavior, respectively, of the real-world entity involved in the interoperation. Specifically:

- $A\epsilon$ stands for the attributes $A\epsilon_1 \ldots A\epsilon_n$ contained in the model of a real world entity and are exposed to other models in the federation.
- $\Omega\epsilon$ stands for the operations $\Omega\epsilon_1 \ldots \Omega\epsilon_n$ defined in the real-world entity model and are available for invocation by external models. Each operation can have parameters such as $p_1,\ldots,p_n$ that can convey the information needed for computation.

Differing views of one real-world entity is shown in Figure 2.6 (Young, 2002).

NOTE: More details can be obtained in the cited reference.

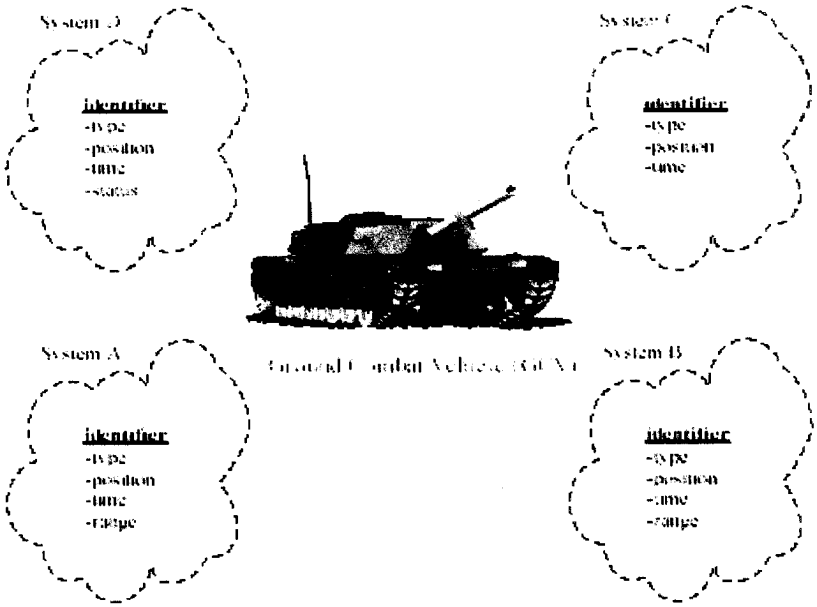

Figure 2.6- Differing Views of an Entity

Different Representation of the same view is pictured in Figure 2.7 (Young, 2002).

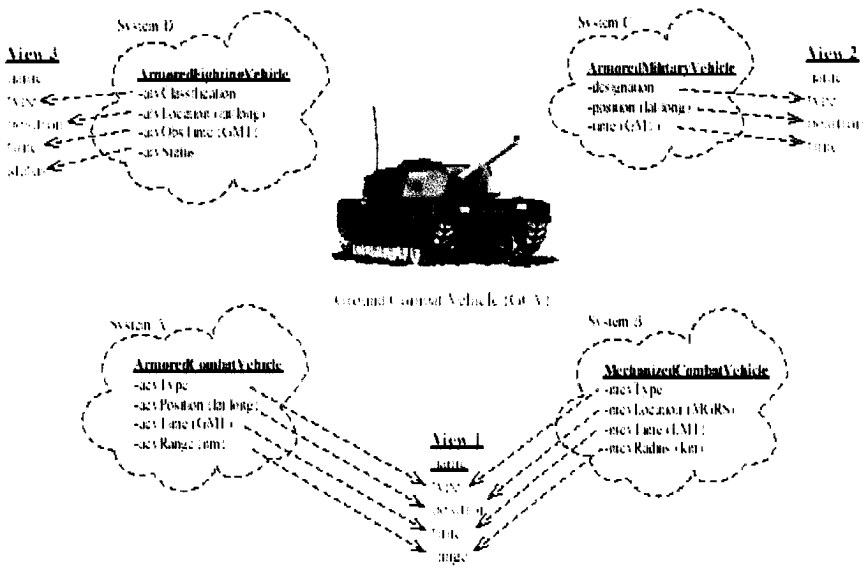

Figure 2.7 Differing Representations of same View

An example of FIOM for a ground combat vehicle is shown in Figure 2.8 (Young, 2002).

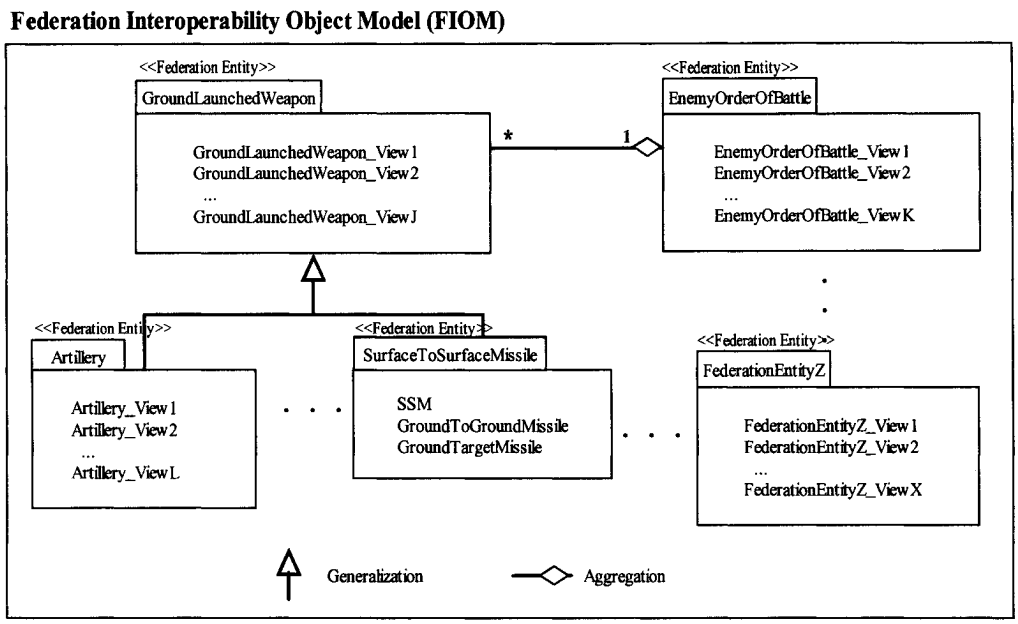

Figure 2.8- Federation Interoperability Object Model

NOTE: Readers can obtain more details on the composition and construction of FIOM by obtaining the cited reference.

Translator

The ultimate task of resolving the differences in views and representations of real-world entities by federation systems lies with the translator. The interoperability object model constructed prior to runtime will be used by the translator during runtime to resolve the differences. The translator will be the intermediary between subsystems. It can be implemented as a *Wrapper*, or as standalone '*hub*' between subsystems.

Figure 2.9 shows an overview of translator as software wrapper and its relationship to FIOM (Young, 2002).

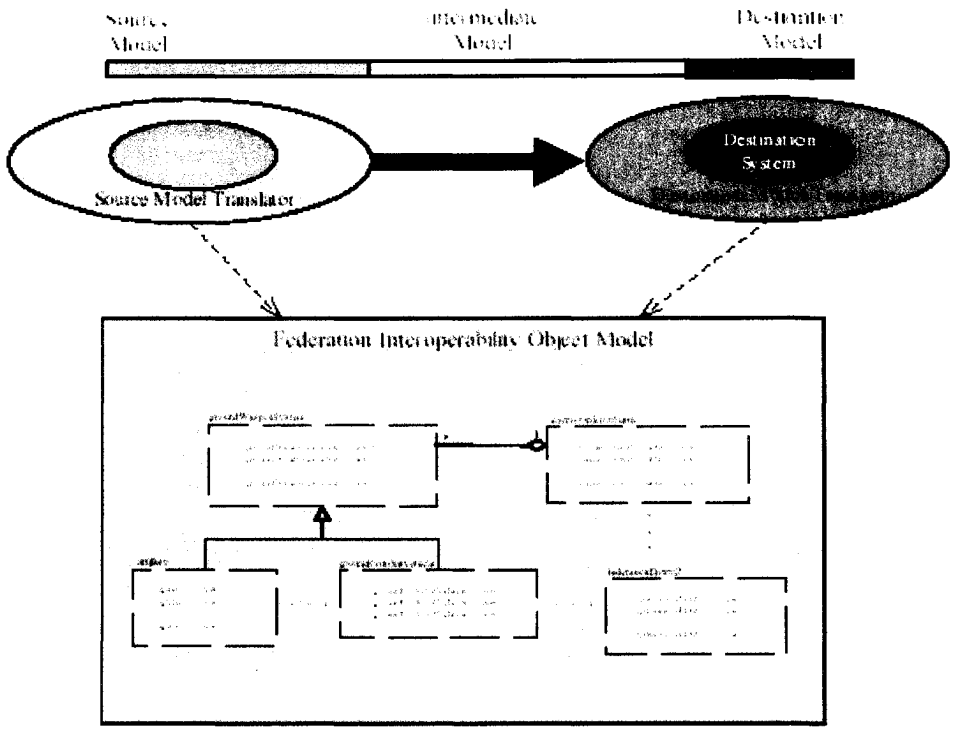

Figure 2.9- Translator-FIOM Interactions

For information exchange, the source system provides the information in the form of a set of attributes or objects in its own format. This information will then be converted into the expectant format, which is the format that the receiving system expects. For joint task operations, the method of translation is the same.

NOTE: Readers can obtain more details on the Translator by accessing the noted reference.

2.2.3 Holistic Framework for Software Engineering

Holistic Framework for Software Engineering (HFSE) provides the collaborative mechanism (Puett, 2003) needed for heterogeneous software development tools and models to interoperate. This framework resolves the compatibility issues, as well as synergy "inadequacy" of software development tools that strive to improve individual aspects of their software development in the spirit of competition.

HFSE is both a framework (a conceptual one) and a methodology that can be used to trace the dependencies, both the type and the degree of dependency, of software development artifacts. The HFSE is also used to quantify and deploy the artifacts via middleware. HFSE enables this overarching collaborative capability by integrating the Quality Function Deployment (QFD) with the Relational Hypergraph (RH) Model of software evolution (Puett, 2003). This integration

allows the system dependencies to be automatically tracked throughout the development cycle, and to document decision-making criteria and details for future engineers in modification efforts.

The potential benefit of HFSE to system safety is the basis for using this framework in this research. The potential benefits stem from the fact, that a holistic model will be able to identify the safety critical software and their effects automatically more reliably and quickly than an engineer, and the dependencies of software functions with safety impact can be traced throughout the development domain.

HFSE is the holistic framework that is established by embedding the relevant portions of QFD methodology into the already existing Relational hypergraph Computer–Aided Software Evolution model, then integrating this extended evolution model into FIOM created from tools and models used by development team.

The holistic framework can be viewed as an abstract layer of activity that interacts with subordinate software development tools via middleware communication tools. Figure 2.10 depicts this framework.

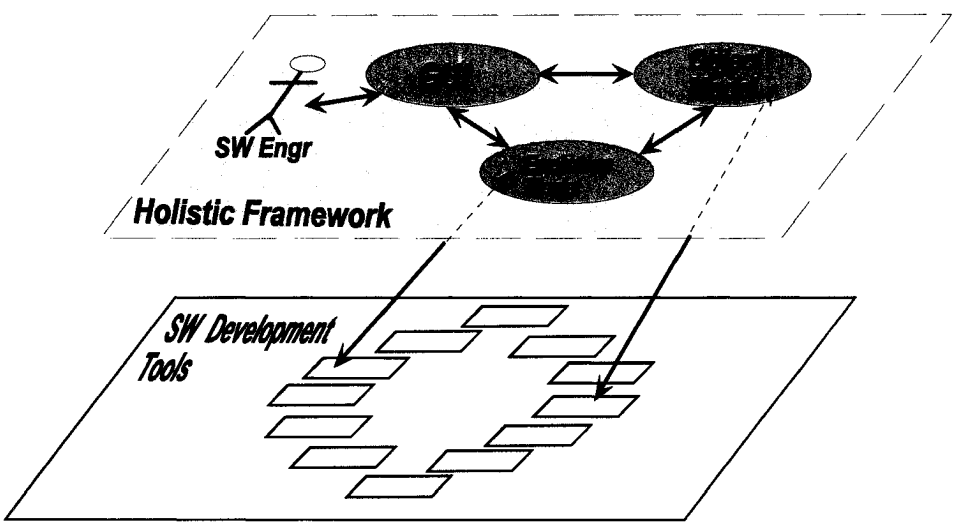

Figure 2.10- Holistic Model of Software Process Interaction (Puett, 2003)

In this section, an overview of HFSE will be provided, however, for more details into the QFD and RH, it is recommended to obtain the cited reference.

HFSE Components

a. QFD- System Requirements account for one third of program risk associated with developments. The problem range from ambiguity of requirements to proper implementation of them. Lack of user input during the requirement development is also a significant problem area. Standish group (1994) (Puett, 2003) reports the three most experienced factors that can cause challenges for software development. These are: 1) lack of user input: 13% of all projects, 2) incomplete requirements and specifications: 12% of all projects, and 3) changing requirements and specifications: 12% of all projects. These factors produce the most errors in design, and for DoD, the changes and fixes can be very costly. QFD is a requirement-based methodology that is designed to overcome the above problems; namely, 1) to ensure that the user's requirements and input are captured throughout the development phase, 2) to ensure that there is no loss of development information (this is important feature since software engineers come and go), and 3) to ensure that different teams in the development are in concert with each other in developing what customer has required.
QFD, integrated in the software development evolution process, enables the software engineers to have visibility over all aspects of development effort, and allows them to work in a more coordinated, quality-based spirit and to make better decision and produce better products.

Last but not least, the QFD has been a successful methodology used to adequately manage safety critical requirements for complex space systems (Dean, 1992) by ensuring that safety requirements are adequately and completely deployed throughout the life-cycle development.

b. RH
The relational hypergraph model for computer-aided software evolution model was established by Harn (Harn, Berzins, Luqi, and Kemple, 1999). This model establishes dependencies and links between software artifacts. In relational hypergraph, activities and artifacts affected by the software evolution are called objects and consist of "steps" and components. It uses a hierarchical refinement (top-level objects, refined objects, atomic objects) to link these objects and establish dependencies (Puett, 2003).

Criteria for dependency are recorded within the object attributes. For example, step attribute may consist of version and variation number, status, predecessor, priority, deadline manager, and evaluator. Component attribute may have version and variation number, hypertext, code, data, pictures, and charts (Harn, Berzins, Luqi, and Kemple, 1999).

Definitions in RH Model

Definition 1. (Hypergraph) (Harn, Berzins, Luqi, and Kemple, 1999) A *(directed) hypergraph* is a tuple $H = (N, E, I, O)$ where

1. N is a set of *nodes*,

2. E is a set of *hyperedges*,
3. $I : E \rightarrow 2^N$ is a function giving the set of *inputs* of each hyperedge, and
4. $O : E \rightarrow 2^N$ is a function giving the set of *outputs* of each hyperedge.

Definition 2. (Evolutionary Hypergraph) (Harn, Berzins, Luqi, and Kemple, 1999).

An *evolutionary hypergraph* is a labeled, directed, and acyclic hypergraph $H = (N, E, I, O)$ together with labeling functions $L_N : N \rightarrow C$ and $L_E : E \rightarrow A$ such that:

1. The elements of N represent unique identifiers for software evolution components,
2. The elements of E represent unique identifiers for software evolution steps,
3. The functions I and O give the inputs and outputs of each software evolution step, such that $O(e) \cap O(e') \neq \varnothing \Rightarrow e = e'$
4. The function L_N labels each node with component attributes from the set C, including the corresponding version of the software evolution component;
5. The function L_E labels each edge with step attributes from the set A, including the current status of the software evolution step, such that $A = \{s, d\} \cdot A'$ (that is, each element of A has the form (s, a') or (d, a'), where $a' \in A'$).

Definition 3. (Relational Hypergraph) (Harn, Berzins, Luqi, and Kemple, 1999).

An evolutionary hypergraph $H = (N, E, I, O)$ is called a *relational hypergraph* if and only if for every hyperedge e in H and every input node n in $I(e)$, the relationship between n and e is *primary_input* or *secondary_input*.

Definition 4. (Primary and Secondary Dependency) (Harn, Berzins, Luqi, and Kemple, 1999).

If an input node and an output node to an evolutionary hyperedge that are different versions of the same component exist, then the path from the input node via the hyperedge to the output node of the step is called a *primary-input-driven path*, and the relationship between the input node and the step is called a *primary_input* dependency. If an input node and an output node of an evolutionary hyperedge exist that are different components, then the path from the input node via the hyperedge to the output node is called a *secondary-input driven*

path, and the relationship between the input node and the step is called a *secondary_input dependency*.

Definition 5. (Top-Level Evolution Step) (Harn, Berzins, Luqi, and Kemple, 1999).

Let $H = (N, E, I, O)$ be an evolutionary hypergraph. A hyperedge $e \in E$ is called a *top-level evolution step* if and only if the hyperedge e has no parent evolution step.

Definition 6. (Atomic Evolution Step) (Harn, Berzins, Luqi, and Kemple, 1999).

Let $H = (N, E, I, O)$ be an evolutionary hypergraph. A hyperedge $e \in E$ is called an *atomic evolution step* if and only if the hyperedge e cannot be expanded to additional steps and its output set has at most one component.

Definition 7. (Top-level Evolutionary Hypergraph) (Harn, Berzins, Luqi, and Kemple, 1999).

A *top-level evolutionary hypergraph* is an evolutionary hypergraph $H = (N, E, I, O)$, each of whose hyperedges is a top-level evolution step.

NOTE: Readers are invited to refer to the cited reference (both HAR99 and PUE03) for more detailed information on hypergraph models.

c. FIOM

FIOM was extensively discussed in Paragraph 2.2.2. For HFSE, the heterogeneity of scope and representation are two important points that are directly applicable to mapping multiple software engineering tools to each other. HFSE uses OOMI FIOM resolution of different levels of abstraction for information provided in different tools and models.

2.3 SYSTEM SAFETY

2.3.1 Objective

The objective of system safety is (USN, 2000) to achieve acceptable mishap risk through a systematic approach of hazard analysis, risk assessment, and risk management. This objective is achieved by getting involved early in development in order to influence the design of the system and eliminate safety critical issues from ever becoming part of the design.

Safety critical functions or issues refer to those functions, lines of code, hazardous conditions that can contribute to death, injury to friendly forces, or damage to assets, equipment, and the environment.

2.3.2 Authority

- **DoD Directive 6055.9** --Requires services to maintain an explosives safety program (USN, 1986).

- **NAVSEAINST 5450.117**- Assigns Technical Authority to NOSSA (WSESRB) (USN, 1986).

- **Section 172 of Title 10, United States Code**--Requires DoD to establish and maintain an explosives safety Program (USN, 1986).

- **OPNAVINST 8020.14/ MCO P8020.11**- Explosive Safety Policy and NAVSEA serve as Department of the Navy Technical Authority for Explosive safety (USN, 1986).

2.3.3 Background

The need for designed-in safety goes back to early 1900s. Hanson, back in 1915, wrote:

Forgetfulness, for example, is not a crime deserving of capital punishment; it is one of the universal frailties of humanity. The problem is, therefore, to destroy as far as possible the interrelationship between safety and the universal human shortcomings, which can be done by designing the safeguards on machines and equipment so that if a man's acts are so essential to safety, it becomes mechanically necessary for him to perform this act before proceeding with his task (Hansen, 1915).

WWII reinforced the need for increased effort in safety following the increase in manufacturing heavy machines, and the associated accidents. The loss of human lives and equipment were so high that the Allies were in danger of losing the war (Leveson, 1999).

Ted Ferry (Ferry, 1984) wrote:

Not so well known is that for a while, accidents in the workplace were nearly negating the increased production. We were losing twice as many aircraft to training accidents as in combat., worldwide. In nearly all theaters of operation, ground and air accidents were three times those of the combat losses .

The effort to design safeguards in the system was abandoned when the war ended.

Post WWII safety approach was called, "fly-fix-fly". An aircraft would be designed and flown until problems surface (or until crashed), then the problems would be fixed, and then flown again.

But with nuclear weapons and space travel, the accidents were going to be catastrophic, and thus this fly-fix-fly or trial-and-error evolved to a more focused, early safety program (Stephans, 2004).

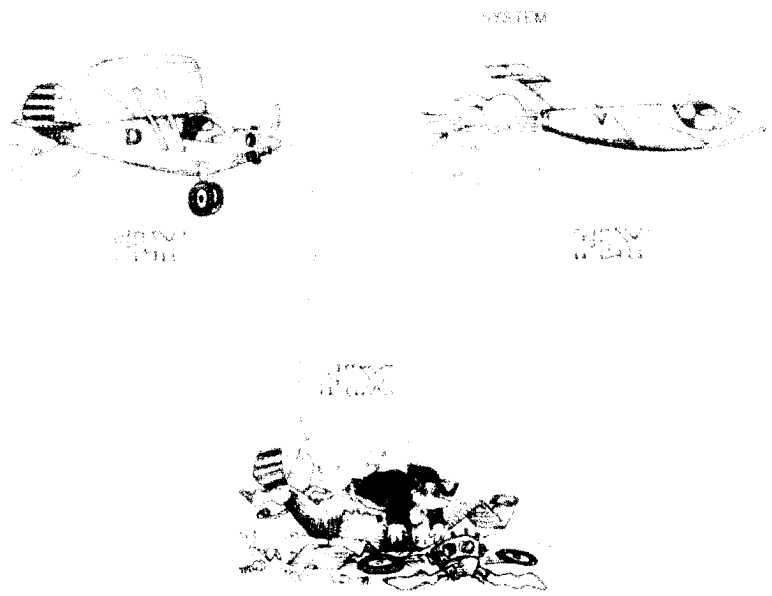

Figure 2.11- FLY-FIX-FLY Approach

The first formal system safety program was established for the US Air Force development of Intercontinental Ballistic Missile (ICBM) in 1962. In 1963 US

Air Force published the MIL-S-38130, "general requirement for Safety Engineering of Systems and Associated subsystems and equipment." DoD adopted MIL-S-38130 as MIL-S-381308A in 1966; then it revised it and designated MIL-STD-882B, "System Safety Program Requirements" in 1969.

System Safety requirements of the Air Force was the basis for DoD's MIL-STD-882 A and B. At this writing, DoD was using version D for its programs. NASA later followed the DoD's system safety approach with a little modification. NASA, however, did little to promote the integration of system safety into overall systems engineering discipline, as MIL-STD-882 strongly requires. The engineers, including safety engineers attempted to bring attention to the Shuttle Challenger's O-ring problem to prevent a catastrophe with no avail. Feynman (Leveson, 1999), in his, *An Outsider's Inside View of the Challenger Inquiry* wrote:

"We saw that NASA had no system for fixing the (Shuttle O-ring) problem, even though engineers were writing letters like, 'HELP!' and 'This is a RED ALERT!' Nothing was done."

In 1990's many other safety documentation emerged. In 1993, the System Safety Analysis handbook was published and is now sold in more than 35 countries. Journal of system safety came about 1999.

In the year 2000, the MIL-STD-882D replaced the "C" version to account for acquisition reform changes and to allow flexibility in implementation or process while preserving basic system safety requirements. This version is titled, "Standard Practice for System Safety." The approach contained in this document will be explained in paragraph 2.8.1.

2.3.4 System Safety Engineering

For us, the indisputable lesson of Chernobyl lies in this: the principles regulating the further development of the scientific-technological revolution must be safety, discipline, order, and organization. Everywhere and in all respects, we must operate according to the strictest standards.

-Mikhail Gorbachev

System Safety Engineering (SSE), an integrated part of Systems Engineering (SE), has at its goal, the early detection of hazards so that the controls can be designed into the system. Figure 2.12 (Alborzi, 2004) depicts the integration of SSE into the SE for optimum design-in safety into the system.

The System Safety Engineering process includes defining the tasks to be performed based on the nature and complexity of the system being developed.

The tasks are different depending on the phase of system development, but the early involvement of SSE team is essential. The typical tasks associated with a complex system is:

- Develop the System Safety Program Plan
- Identify potential top-level system hazards
- Establish a Hazard Tracking Database and a Closed-Loop Hazard Tracking Process
- Identify safety critical components, subsystems and/or interfaces
- Identify Safety Requirements from the system requirements specification
- Perform Safety Requirement Analysis
- Perform Design Analysis
- Perform Code Analysis
- Review and influence the System Test Plan
- Develop safety test procedures
- Verify Safety Requirements via testing
- Review test results
- Document all findings
- Determine the overall residual risk
- Prepare Safety Assessment Report (SAR)

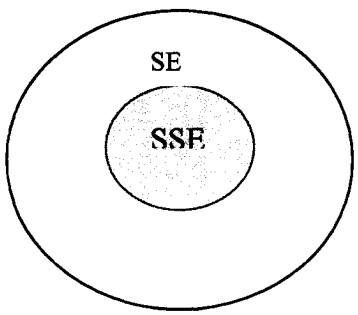

Figure 2.12- SSE- an Integrated part of SE

2.3.5 System Safety Methodology

Currently, there are multiple, but similar SSE processes used for US Navy systems. Each process must comply with MIL-STD-882, NAVSEAINST 5000-2R (DoD, 2003), or other contractual safety standards.

2.3.6 System Safety Analysis

The following are typical analyses performed in a system safety program:

- Preliminary Hazard Analysis (PHA)
- Sub-System Hazard Analysis (SSHA)
- System Hazard Analysis (SHA)
- Operating and Support Hazard Analysis (Q&SHA)

Safety Analyses are performed to study the root causes of hazards in the system as are associated to hardware, software or human, to identify or recommend design feature for controlling hazards, and to establish a relationship of root causes to actual mishap.

The above analyses are performed in different phases of system development and their output is served as input to further analyses. Other analyses such as Safety Requirement Criticality Analysis (SRCA) or Software Hazards Analysis are also performed on some programs.

Figure 2.13 shows the relationship between safety analyses and the risk of mishap.

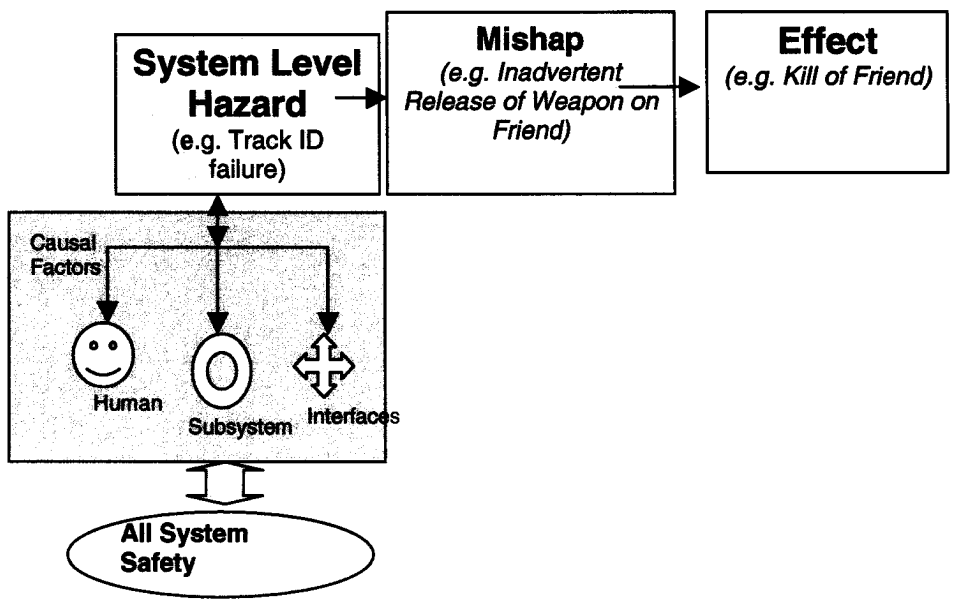

FIGURE 2.13- Safety Analyses and Risk of Mishap Relationship

2.3.7 Analysis Tools

Fault Tree Analysis

Fault Tree Analysis (FTA) is the most typical and oldest method for analysis. It was developed in 1962 by H.A. Watson at Bell Telephone Laboratories to

evaluate the Minuteman Launch Control System for an unauthorized (inadvertent) missile launch (Leveson, 1999). FTA is a top-down analysis of an undesirable event. It is the means of identifying the causes of hazards. FTA uses Boolean logic (and /or) to show the combination of individual faults that can contribute to the top-level event. As the tree branches out and down the tree, the details of the root causes are more evident. Both quantitative and qualitative analysis can be done with FTA. If the probabilities of occurrences of the events on the tree are known, then the frequency of the top event can be determined.

Figure 2.14 shows an example of a fault tree for an inadvertent launch (Alborzi, 2004).

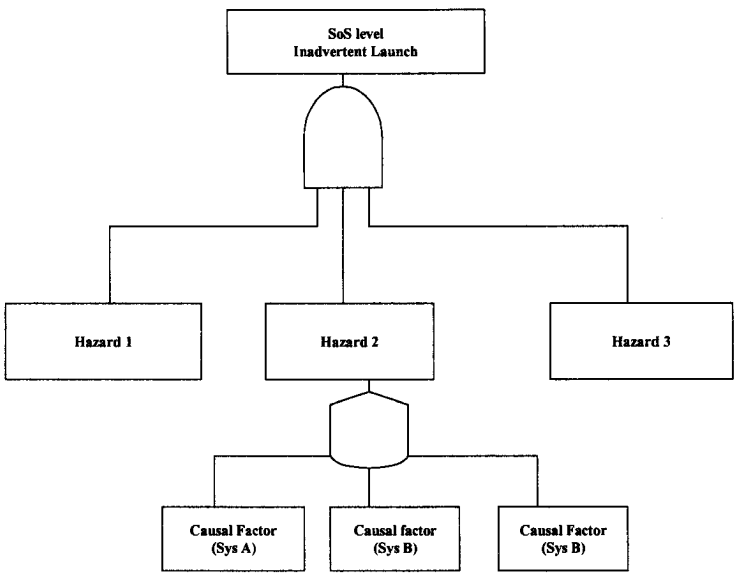

Figure 2.14- Sample Fault Tree Analysis

Failure Mode and Effects Analysis

FMEA is one of the most used analysis tool in safety and reliability. The tool allows the engineer to know what can go wrong with each individual piece of the hardware (Stephans, 2003).

FMEA takes a system and breaks it down to components and then looks at how each individual piece can fail and the effect of each failure on the component and on the overall system. Some of the system failures can be both safety related and reliability related (Alborzi, 2003). For example, system crashes can decrease the reliability of the system, and can also pose safety risk to the assets, equipment or cause the corruption of safety critical data. That is why the FMEA and FMECA are used for both groups.

Failure Modes, Effects, and Criticality Analysis (FMECA)
This analysis adds "criticality" to the Failure Modes & Effect Analysis that is used widely in reliability world. The criticality is expressed in probabilities or frequencies (Leveson, 99). FMECA is the reverse of FTA, that is, it is a bottom-top analysis. FMECA will point out interface dependencies and requirements. Samuel Keene, (Keene, 1992) gives an example the module will be susceptible to conditions from which it cannot recover without an outside system. FMECA is the right tool for identification of these dependencies.

Energy Trace and Barrier Analysis
Energy Trace and Barrier Analysis (ETBA) starts with identification of the types of energy associated with the system. For each energy type, the points where the energy enters is identified and the path of energy flow is traced throughout the system. The next step is to identify and evaluate the barriers that are in place to control the flow of energy. The evaluation includes the identification of potential damages in the event of barriers failure. Next, the risk associated with potential damage (as the result of unwanted energy flow (Stephans, 2004), will be determined, and expressed in term of risk assessment code (RAC).
The last step is optional and that is the recommendation for improved barriers/controls.

2.3.8 Risk Assessment

Risk is measured by merging the effects of two dimensions- the *probability* of occurrence of a mishap, and the *severity* of the same mishap, if it were to occur.

The *Probability* dimension has five levels from "frequent" to "improbable" based on the criteria chosen in MIL-STD-882 (USN, 2000).

The second dimension is *Severity*. There are four levels of severity from catastrophic to negligible, and the associated criteria (USN, 2000).

Mishap risk is the merging of probability and severity. An example is shown in Table 2.2 Table 2.3 shows the authority for acceptance of risk.

Table 2.2- The Mishap Risk Index

MISHAP PROBABILITY	**MISHAP SEVERITY CATEGORY**			
	I- Catastrophic	II-Critical	III-Marginal	IV- Negligible
A- Frequent				
B- Probable				
C- Occasional				
D- Remote				

E- Improbable				

Table 2.3- The Risk Acceptance Authority

CELLS	RISK LEVEL	RISK ACCEPTANCE AUTHORITY
IA, IB, IC, IIA IIB	HIGH	Acquisition Executive
ID, IIC, IIIA, IIIB	SEROIUS	PEO IWS
IE, IID, IIE, IIIC,IIIE, IVA,IVB	MEDIUM	Program Manager
IVC, IVD, IVE	LOW	

2.3.9 SOFTWARE SAFETY

Process Alone Not the Measure of Success

A small number of software components are responsible for a disproportionately large number of faults in any large-scale system (Basili, Perricone, 1984). Most of the faults occur are safety related. Chen (Chen, Lang, 1995) notes that accidents usually involve a complex interaction of incidents with multiple contributing product, process, and people/resource (P3R) factors. This implies that we need to find a proper balance of emphasis on P3R issues. Chen's P3R delineation is also illustrated in IEEE standard taxonomy for software engineering standards (ANSI/IEEE, 1992). In the Acquisition reform era, in DoD, it has been a tendency to emphasize one of the element usually the process, over the other elements. This will lead to compromising safety. Littlewood (Littlewood, 1993) says it correctly: "There is almost no empirical evidence to confirm that process-based standards alone can ensure safety."

Hamlet and Voas (Hamlet, Voas, 1993) state that there is no direct correlation between process and quality of work, specifically,

"All of these ideas, from process definition and control to systematic testing have one failing in common: there is no established relationship between the method and quantitative assessment of the quality that methods is supposed to engender."

In the early 1990s, the Institute of Electrical and Electronics Engineers, Inc., (IEEE) Software Engineering Standards Committee (SESC) formed the Software Safety Working Group. The group's charter was to develop a software safety standard that described, "The minimum acceptable requirements for the content of a software safety plan...which is used for the development procurement,

maintenance, and retirement of safety critical software (IEEE, 1994). The result was IEEE Std. 1228 and was approved in March 1994.
This standard focuses on software safety issues, and recommends that a software safety plan be developed as a part of overall system safety program. The goal of this standard is, "to address the processes and activities intended to improve the safety of safety critical software"(IEEE, 1994).

Software and Uncertainties
Recently, the need to re-use reliable software libraries to interface with legacy systems has surfaced few interoperability issues. These issues easily propagate to the rest of the systems involved in interoperation. In U.S. Navy systems, when the mission involves the risk of human (Gill, 1991), evaluation of the safety of the software controlling the system is required. Currently much of the evaluation is done manually, therefore costly and error-prone. Fault Tree Analysis done traditionally for the hardware, is now being applied to software (Leveson, Harvey, 1983). The Fault Tree Analysis has been successful since its early development in 1960s, when first was used for Minuteman missile system (Hammer, 1972).

With the rapid migration to systems of systems architecture, the software architecture, that is, the high level description of its computational elements, the means by which they interact (Gamble, 2001), and the structural constraints on the interaction (Perry, 1992), is of great interest to safety engineers. The software re-use and software modification in safety critical code have brought new challenges for safety engineers. Several different computer languages, data types and requirements are required, for systems in the federation, to be interoperable without advert safety implications. The solution is usually bringing new middleware to provide a generic and reusable solution to communication conflicts (Gamble, 2001). However, the inherent uncertainty associated with software in safety critical systems and reconfiguring the systems, becomes a "certain" risk that cannot be quantified. If risk could not be quantified, it would be difficult, if not impossible, to reduce or manage it.

(Leveson, 1999) points to the *myth* associated with software- a belief that software cannot "fail" and that all errors will be removed by testing. On the contrary, software can fail and the consequence can be described in accident like Arian 3 explosion, as the result of dead code being executed, and MARS probe loss as the result of simple unit of measurement discrepancies.

Unfortunately, it takes catastrophic accidents, for the managers and stakeholders, to pay attention to software safety and to fund it adequately. One of the main reasons why software safety analysis is not funded adequately and as the result, not performed well, is the lack of good evidence or data in the benefits it provides. The benefits are long term and usually come indirectly, in the form of no-accident. Most governmental agencies or industrial agencies are more concerned with short-term goals.

2.3.10 Safety Review Boards

The Navy programs are evaluated, at each critical phase of the development lifecycle, by a formal, independent review board (s).

- Weapon System Explosives Safety Review Board (WSESRB)

The WSESRB (NAVSEA, 1997) has responsibility for independent oversight of the safety effort associated with all ordnance items, weapon devices, or systems used, handled, stored or tested aboard a Navy ship or aircraft. The WSESRB functions by providing safety-related guidance and recommendations to acquisition programs and Milestone Design Authorities (MDA). The WSESRB also provides recommendations to MDA regarding safety approval of designs entering low-rate initial production (LRIP) and/or prior to final production approval for weapon systems, related systems and materials, and weapon system software considered for use by the Navy.

The WSESRB consists of representatives from the Naval Systems Commands and other Naval activities as deemed appropriate by the chairperson (NAVSEA, 1997). WSESRB members may seek assistance in technical reviews and may request attendance of technical experts at WSESRB meetings, as necessary.

The WSESRB, in general, makes safety recommendations to Program Managers (PM) and MDAs (NAVSEA, 1997). In light of compelling requirements, the PM or MDA may decide not to comply with the WSESRB's safety recommendations and to accept an increased level of risk associated with an identified hazard (NAVSEA, 1997).

- Technical Review Panels (TRP)

A subset of WSESRB, these panels examine the software, or other unique aspects of the system. Its recommendations and findings are not final until WSESRB's approval.

2.4 THE "GAP" IN THE LITERATURE

Table 2.2 lists some of the most known and focused literature in areas of:

- System Safety/ Software safety
- Interoperability
- Systems of Systems

2.4.1 The Method for Literature Review

The literature on System Safety and Software Safety were reviewed based on the following 5 criteria:

- Single System

- Hardware
- Software
- Human
- Interoperability

If the document applied to any combination of the above, it was checked in this table.
As evident in the table, none of the safety documents addressed non-single system architecture or interoperability.

Then the literature on Interoperability was reviewed. The following criteria used was:

- Performance
- Training
- Testing
- Safety

If the document applied to any of the criteria or combinations of the criteria, it was marked. As evident, none of the literature in this subject addressed "Safety".

Then, the literature in the area of systems of systems was reviewed against the following criteria:

- Performance
- Training
- Testing
- Safety

The pool is narrow in this area, but there were enough to conclude that system safety was not in the mind of the authors.

Table 2.4 shows clearly that the body of knowledge lacked in one area: Safety Interoperability. This is the focus of this research and analysis.

TABLE 2.4- The "Gap" in the Body of Knowledge

	AUTHORS	Literature on System Safety					Literature on Interoperability			
		HW	SW	Human	Single SYS y/n	Inter-operability	Performance	Training	Testing	Safety
1	Herrmann		X		Y					
2	Leveson, N.	X	X	X	Y					
3	Shimeal/ Gill		X		Y					
4	Roland	X	X		Y					
5	Stephans		X		Y					
6	MIL-STD-882	X	X		Y			X	X	
7	Haimes	X	X		Y					
8	Puett				N		X			
9	Young				N		X	X	X	
10	Scultz/DISA						X			
11	Benkhallat/Siebert								X	
12	Davis/Payton						X			
13	Kaplan/ Wileden						X			
14	Hamilton						X			
15	Bubba/Wileden						X			
16	Correa-Martinez				N		X			
17	Carlock						X		X	
18	Kotov						X			
19	Luskasik						X			
20	Manthorpe						X			
21	Keating				N		X			
22	Phadke						X	X		
23	Flood/Carson						X	X		
24	O'conner						X			
25	Bar-Yam						X			
		Literature on Systems - System of Systems								

The "Gap": Safety Interoperability

2.5 SYSTEM SAFETY INTEROPERABILITY- A NEW CONCEPT

Up to now, interoperability has been viewed as a "performance" issue (even by some safety engineers), and so it is often thought of being outside the scope of system safety. Many books and technical papers have been written to explore the interoperability aspect of engineering. Some *interoperability-friendly* tools such as Java, CORBA, have been successful in facilitating the integration effort between multiple complex systems.

The magnitude of the software safety challenge in today's interconnected and interrelated systems is before us every time we rely on air traffic controllers to get us safely from one part of the continent to another part while thousands of other airplanes are doing the same thing, at the same time in various directions. Some organizations have started to grasp the necessity and the complexity of safety as the intrinsic property of interoperable systems. The FBI has recently established a National Infrastructure Protection to protect safety-critical systems and software (Hermann, 1999). Department of Defense has initiated a broader, more system-of-systems-based approach in embedding safety features in the design of the overall system architecture.

In U.S. Navy, the idea of a "combat system" in engineering domain (minus the system safety) is a familiar term. The integration issues are dealt with and resolved. But system safety analyses were always conducted on single systems and based on single system view. This practice continued until USS NIMITZ was ready to sail as a part of the battle group and needed its Ship Self Defense System (SSDS) to get approval from Weapons Systems Explosives Safety Review Board (WSESRB.) The WSESRB did not concur with the safety effort on the various systems until a new comprehensive, system of systems level safety effort was established.

Consequently, in 2002, USS NIMITZ (CVN 68) Ship Self Defense Combat System Safety Program launched, for the first time, an initial step toward analyzing a combat system (i.e. system of systems) for safety. This "ad hoc" approach to system safety was completed when USS NIMITZ sailed in 2002 only few months after the safety program initiated (NAVSEA, 2002). More detail on USS NIMITZ is provided in 2.8.2 as it is used as an approach for analyzing safe interoperability of complex system of systems against the safety interoperability criteria in paragraph 2.7.

In 2004, USS REAGAN initiated a more thorough combat system safety analysis (NAVSEA, 2004). Lessons-learned from USS NIMITZ helped in the preliminary analysis process. But the program does not have the benefit of a structured and formalized methodology. There is not a model other than USS NIMITZ to follow. This effort began earlier and used USS NIMITZ lessons-learned (NAVSEA, 2004). But interoperability issues from safety perspective have not been analyzed

in depth, partially because no framework exists as of yet. Lack of system of systems safety engineering methodology precluded, up to this day, the in-depth analysis of interoperability issues or even risk assessment. This program is still under work and it is in its early lifecycle. The approach used for this program is identical to USS NIMITZ, but with more time. Since this safety program is in its early phases, and can add no new method /perspective for analysis, it will not be used in our comparison and evaluation against safe interoperability criteria.

2.6 SYSTEM SAFETY INTEROPERABILITY ISSUES

As explained in paragraph 1.5, safety is "Freedom from those conditions that can cause death, injury, occupational illness, or damage to or loss of equipment or property, or damage to environment" (USN, 2000).

The definition above is an absolute. In practical sense, we will never have "freedom" from those hazardous conditions in a complex system. Safety issues refer to those conditions, entities, and hazards in the system that, given a "right" circumstance, can contribute or lead, directly or indirectly, to an accident. Inversely, an issue is safety- related when it can cause a hazardous condition in which an undesirable event may occur.

2.6.1 System Safety Interoperability Impediments

Gamble (Gamble, Davis, Payton, 2001) gives a big portion of "credit" for interoperability conflicts to software architectural issues. He disputes some of the DoD's steps toward post-integration assessments, mainly because once the problems are discovered, there is no time to fix them. He proposes pre-integration assessment of the to-be system, mainly based on its software architecture. An integrated architecture is a software architecture description of the overall solution to interoperability problems between all interacting systems. The objective of pre-integration conflict assessment is to choose the right middleware.

The requirement for interoperability between complex systems that were developed independently, without the requirement for interaction, introduces a few challenges. How do we resolve the different computer languages used in these systems? How do we resolve modeling differences in data used in these systems?

As shown in Figure 2.6 Young (Young, 2002) explains that the modeling problem is one of the biggest impediments to interoperability. Each system in the federation of systems has a different 'understanding' or a 'model' of a tank. Some of these modeling differences used in the Navy's complex systems also carry great safety implications. We need to understand the sources for these differences and learn how to mitigate them in the system design.

The root causes for differences is explained in paragraph 2.2.1. The paragraph also lists and describes the several heterogeneities that have been noted by the researchers. Here, we choose those heterogeneities that carry safety implications and focus on *safetyness* of the heterogeneities.

Heterogeneity of Hardware and Operating Systems

These are differences in operating systems and the hardware used when new, legacy or hybrid complex systems are integrated. The speed in technology advances makes this problem a certain one in a federation of systems. The hardware platform differences include how the same information is displayed in different systems, such as the size, length, and format of the data. The differences in operating systems have been a more "familiar" problem. In DoD, the two most utilized operating systems in the world of COTS are Microsoft Window and UNIX products.

Heterogeneity of Structure

This modeling difference refers to differences in structural composition, schema, or implied vs. explicitly stated information. Young, (Young, 2002) explains that this difference can arise when a real-world entity is modeled as an object on one system and as attribute in another, e.g. such as an aircraft route being modeled as an attribute of an aircraft mission object on one system (as one element of the overall mission) and as a separate entity used for de-conflicting missions in another (Holowczak, 1996). It will be shown later that "de-confliction" is a safety critical issue.

The heterogeneity of structure also refers to concepts being modeled differently in the schemas of corresponding systems, e.g. a relationship that is modeled as one to one in one schema and one-to many in another (Hammer, McLeod, 1999). This type of mismatch is especially apparent in IFF interrogation tasks. IFF processing is noted later in this section as a safety related issue.

Heterogeneity of Presentation

This heterogeneity includes domain mismatch problems. For example, the use of different units of measure, difference in precision, disparate data type, word size and constraints integrity. From this list only precision is not safety critical. Domain mismatch occurs problems occur when the same concept is characterized differently in two separate systems (Young, 2002), such as geographic position measured in latitude and longitude on one system and Military Grid Reference System (MGRS) on another (Hammer, McLeod, 1999). Another example is for a system to use meter as a unit for distance measurement, and another system to use yards. Disparate data types refer to expression of the same characteristic in different data types, for example, using a telephone number as an integer in one system and as a character string in another (Young, 2002).

Heterogeneity of Temporal Validity

This modeling difference arises from differences in the time used by two models to observe or record the state of a real-world entity (Young, 2002). These temporal validity issues are particularly an issue with military C4I systems (Holowczak, 1996, Wiederhold, 1993). For example in one C4I system, the satellite picture of a threat area may be kept for a month to be valid (and be used), and the same picture in another system may kept for a year to be valid (and used). Validity of this real-world entity may have further implications depending on how it is used.

2.6.2 Top System Safety Interoperability Issues

Issue of Identification Friend or Foe (IFF)

Identification systems are used in every aircraft and every ship, civilian or military. Pre-set codes within various modes are agreed upon between the friendly units (Ikram, 1998). The unit is challenged, and if the incoming unit responds correctly to the pre-set codes, it is considered a friendly. If not, the unit is considered "hostile" and further actions will be taken against this unit.

The safety issue arises when a discrepancy in identification, that is mis-identification or no identification, can lead to prosecution of a unit that may very well be a friendly. Unofficial DoD estimate is that 24% of U.S. fatalities in 1990-91 Gulf War were caused by friendly fire. Reed (Reed, Little, 1992) states that in Dessert Storm experiences involving highly mobile close support operations, were one where the inability to IFF ground targets such as men and small vehicles caused several incidents of fratricide. Different and multiple identification systems used in the theater also accounts for some safety issues. The identification given to a target by system X and passed to system Y, does not match with the identification on the same target processed by system Y. Cooperative Engagement Processor's (CEP), a component of Cooperative Engagement capability (CEC), main task is to correlate track ID's and track IFF modes and to pass that final-correlated, comprehensive view of a target to all requesting systems. GEO DB information is used in CEC for situational awareness and composite identification function. Potential ID differences exist if one system has a different version of DB than another system in the combat system (CS) or in other platforms. Potential engagement of a Friend or Non-Hostile may occur.

Recall paragraph 1.3.1, "6 April 2003; an USAF A-10 kills one British soldier and injures several in a friendly fire incident in southern Iraq" (CALL, 2004)."

Also, the U.S. Navy has recently implemented an upgrade in its AEGIS computer programs to account for modification made in UPX-XII.

The Interrogator System, AN/UPX-29(V) is an Identification Friend or Foe (IFF) System compliant with requirements for an Air Traffic Control Radar Beacon System (ATCRBS) IFF Mark XII System (AIMS). Mark XII requirements have been modified to include Mode 5 capabilities that together with the current Mark

XII requirements make up Mark XIIA. The AN/UPX-29(V) will be upgraded to provide Mark XIIA (Mode 5) and Mode S capability. The secure Mode 5 employs a new waveform, time dependent interrogations, and new cryptography. It also provides the capability of transponders to downlink to the interrogation platform the standard Mark XII Selective Identification Feature (SIF) modes, a unique Platform Identification Number (PIN), National Origin (NO), and Mission Codes (MC). Mode S is the civilian upgrade to the standard Mode 3 and C interrogation and replies. Mode S provides the capability to selectively request and receive specific register data such as Global Positioning System (GPS) position, aircraft type, and VHF radio frequencies for an aircraft.

Additionally, included in the changes is the expansion of Mode 1 from two-digit octal to four-digit octal. It is called extended Mode 1.

In order to maintain interoperability between the Mode 5 and Mode S capable Aegis ships and non-Mode 5 and non-Mode S participating units over the Link, Command and Decision (CND) will not override an extended Mode 1 code with a Mode 1 code or a non-enhanced Mode 1 codes. Modes 2, 3, and 4 are legacy codes.

Prosecution of a friendly track based on erroneous IFF reply can have a catastrophic consequence. It is apparent that this change, and any other IFF data passed between systems is a safety critical issue in the interoperability requirement.

Inability to distinguish Training/Test Tracks from Tactical Tracks
Recall the accident noted in paragraph 1.3.1,

" 17 April 2002; four Canadian soldiers are killed and eight wounded when a USAF F-16 mistakenly bombed them as the Canadians were engaging in a live-fire training exercise at night" (CALL, 2003).

When complex systems of systems are integrated and interrelated to perform joint execution of tasking, it is imperative that they all are aware when one or more systems are in training, test or maintenance mode and that tracks being tracked and managed are simulated and not real. The separation of training and test tracks from tactical tracks by design enables all link units to distinguished between the two types and not try to engage a track that doesn't exist. Causal factors related to confusion can be either:

- No integrated training capability in one of the systems,
- No effective communication,
- No use for simulated data by a system therefore everything is Real, or
- Residual training tracks after mode transition.

Break Engage / Hold Fire / Cease Fire

Break Engage capability allows operators to stop an unsafe engagement. The engagement is stopped and all engagement related information is cleared from engagement queue.

Hold Fire capability allows the operator to stop the engagement but retain all engagement related information in engagement queue.

Cease Fire capability allows the operator to continue with the operation of a Missile in Flight (MIF) and with a battery-activated missile, but no more firing is permitted.

Inability to break engage or failure to break engage/hold fire can have catastrophic consequences. This is usually due to console failure (loss of VAB Array), loss of interface between systems during engagement. Casual factors such as processing delay and corrupted engagement data can cause this safety issue to occur (Alborzi, 2004).

Engagement Status is a sub-issue where operator confusion may lead to an improper action. These status problems include a) Wrong status of engagement displayed as the result of malfunction of display system or timing between break engage and previous engage status did not allow for correct status display.

System Crash

Tools that observe and manipulate the runtime behavior of parallel and distributed systems are essential for developing and maintaining these systems. But the infrastructure requirements must be in place to support run-time tools, otherwise there is no interoperability or there is an interoperability that causes system crashes or system malfunction. In a war-fighting system, a system crash can be catastrophic due to loss of interface with the weapon in flight.

One of the issues, the "premature launch" (Alborzi, 2004) is explained in detail to show the safety criticality of the issue.

Pre-Mature Missile Launch

There are occasions when a low risk element level hazard may have greater safety implications when considered in the Systems of Systems context. As depicted in the simplified illustration below (Figure 2.15), the Weapon Control System (WCS) interfaces with the launcher system and launcher in turn interfaces directly with the missile.

The WCS has a requirement to issue Flight Target Data (FTD) to the launcher system when requested by the launcher and all validation checks within the WCS are verified. For example, the WCS will ensure that the missile is selected for launch, the proper mode has been achieved, and the message from the launcher is in the proper format and can be processed.

The launcher has requirements to request the FTD when it is required during the launch sequence, validate the data received from the WCS, transmit the FTD to the missile and monitor for a Ready To Launch (RTL) indication from the missile indicating that the FTD has been received and validated by the missile, and the missile has performed a health and status check and is ready for launch. Upon receipt, the launcher then commands ignition of the missile's rocket motor and monitors for a missile away indication.

Missile requirements are to validate and process the FTD received from the launcher. Once FTD is processed and the health status has been determined ready to support the launch, the missile initiates a health and status checkout of the missile systems and responds to the launcher with an RTL.

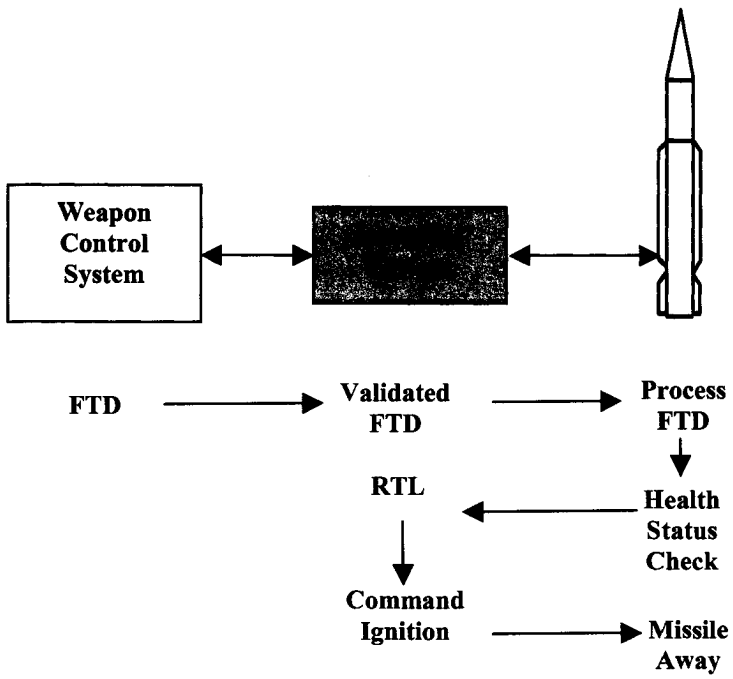

Figure 2.15 – An Example of Safety Interoperability Issue

Now consider that there is a known hazard in the missile system whereby the missile sends the Ready to Launch (RTL) signal before it has fully processed the FTD and performed the health and status check. The hazard assessment determines this is a low risk missile hazard in as much as the impact is negligible and the probability is remote ($10^{-3}>P>10^{-6}$). However, the consequences of this hazard coming to fruition are more significant from a system of systems perspective.

Given the occurrence of the hazard in the missile, that is the missile sends the RTL signal to the launcher before it has fully processed the FTD and performed a health and status check, the launcher will validate the RTL and immediately command missile ignition. This will result in the missile launching prematurely, without having the requisite data to know where it is to fly; i.e. the missile is flying blind. The consequences could be catastrophic.

Another example of safety interoperability issue was provided in paragraph 1.5 to facilitate the understanding of safety interoperability concept.

Table 2.5 summarizes the safety interoperability issues identified as the result of literature review.

Table 2.5- Safety Interoperability Issues

	Safety Issues	Sub-Issues	Causal factor(s)
1	Air Control	Sending the air controlled A/C toward Target. No notification to A/C on imminent firing	Link problems
2	Deconfliction	Operator Confusion Duplication of operation Timing, location codes	Friendly Fire Damage by Target penetration Accidents
3	Distinction of Real Tracks from Simulated	Residual training tracks in tactical mode Misidentifying a training track from Real Inability to distinguish between Tactical and Training tracks	No integrated Training capability No robust communication No use for Simulated data by a system (therefore everything is Real)
4	Engagement /Break Engagement Issues	Engage on Friendly Failure to Break Engage/ Hold Fire Inability to Break Engage/Hold Fire System is in Engaged status, but display doesn't show System not in engage status, but display shows system is	Identification error Display Interface Buffer not cleared Operator error Break engage, hold fire, not working Timing
5	Force Orders	Data integrity	Track data corruption

6	Identification	ID Inaccuracy ID Doctrine ID Differences ID MGMT and Conflict Mode 4 Incorrect No Deconfliction	Doctrine Communication Geographic DB versions Interrogation problems, wrong pre-selected ID
7	Mode Mismatch	Training Tactical Test Maintenance	Inability to communicate No multiple mode capability Display Mode problems
8	Premature Launch	Weapon flying Blind Engage friendly	Timing delay Loss of interface
9	Reference Points Anomalies	Ownship Other areas/tracks	Interface/communication Track data corruption
10	Threat Evaluation	Threat values Threat Evaluation Process IAW weapons doctrine	Auto engagement doctrine can result in release of ordnance w/o operator initiation
11	Track Anomalies	Correlation problems Track Inaccuracy Dual Tracks Tracking Problems Redundant Local tracks Excessive TN Changes	Different systems using different scheme for track numbering Modeling problems Overlay problems Track quality problems

Table 2.5- Safety Interoperability Issues- continued

2.7 CRITERIA FOR EVALUATION OF SAFE INTEROPERABILITY ANALYSIS APPROACHES

In order to elect a systemic analysis approach to evaluate the safety of the interoperation between multiple complex systems, we need to establish a list of criteria for evaluation. By carefully assessing each approach against the criteria, we can optimize the quality of our analysis from system safety perspective. Based on the safety interoperability issues identified above, the author of dissertation has identified the top 6 safety related criteria (A thru F) for evaluation. Each criterion will be discussed briefly in this section.

A: SoS-based
B: Address of Model Correlation
C: Knowledge /Safety Requirements of Remote Operations (interfaces)
D: Feedback (validation &verification) Capability
E: Translation Methodology (messages, representation of entities, etc.)
F: Safe Federation Extensibility Capability

SoS-based: The approach used for analysis must be based on systems of systems context. Single-based approaches are unable to address the safety implications, nature and/or the requirements for interoperation between multiple complex systems that need to be logically and functionally integrated and interoperable.

Address of Model Correlation: The approach chosen for analysis must address modeling differences and resolutions/mitigations for differences. Developing a federation of autonomously developed heterogeneous systems involves many real-world entities and potentially many models of those entities by the many systems (Young, 2002). Manual correlation of different models of each real-world entity could be difficult, unsafe, and time-consuming effort. In integrating database systems, the data correlation problem poses one of the biggest safety issues. Cooperative Engagement Capability (CEC) program uses Geo-base data for track / ID correlation. Safety issues related to track correlation account for top ten percent of CEC's safety hazards.
The candidate analysis approach will be evaluated against this criterion for safe interoperability purpose.

Knowledge / Requirements of Remote Operations (interfaces): An approach to analysis of safe interoperability capability must be able to address interface requirements. First is the knowledge or awareness of the existence or identity of those systems with which it communicates. Generally, filter in the pipe (Gamble, 2001) and filter architectural style are unaware whereas object-oriented architecture is aware. These requirements include knowledge of interfacing

systems' functionalities, a mechanism by which each system in SoS can "talk" and "be understood" by other systems, and receive data. This mechanism enables a system within a federation to invoke operations implemented on another systems. Safe-invoking operations is the criterion for analysis approaches. Safe-invoking approach of safety critical command of other systems in the federation is essential for the safe operation of the overall system of systems. Loss of safety critical interface falls in this criterion. For example loss of uplink data to the Missile in Flight (MIF) is safety critical issue because Missile uses the uplink data to guide itself toward the target. Not having the uplink data as the result of interface loss leads to the MIF flying blindly, therefore causing a catastrophic accident.

The candidate analysis approaches will be evaluated to determine how the safeguards been designed in their remote callings methods (i.e. using a designer who has prior knowledge of other systems operations, or invoking a server's methods using the client's representation for the method name and parameters) (Young, 2002). Also, the approach will be evaluated on the basis of sound and unambiguous safety requirements for interface operations.

Feedback (Validation & Verification) Capability: The next criterion is that the candidate analysis approach must address each system's need and capability to give feedback to other systems in the federation, to validate the safety critical messages as for their format, their content, their size, their acceptable boundaries, etc., and verify that they can be used for operations. It must also address the mechanism by which erroneous information is returned to sender without causing corruption or malfunction of the receiving system. Erroneous information include incomplete messages, formatting problem, or "not-understandable' information. The feedback capability is more than a "hand-shake" capability; it verifies that a message can be processed (used) in the receiving system. The analysis approach must allow for managing interfaces by listing and documenting controls, dependencies, and data objects across the interfaces. Input errors are checked in real time (Keene, 1992).

Translation Methodology: Early interoperability attempts involved the creation of custom point-to-point (also known as source-to-destination) interfaces between systems (Young, 2002). This approach was to resolve representational differences between multiple complex systems, potentially requires n(n-1) translation for a federation of n systems. That is if there were 4 systems in the federation, we needed to have 12 translations.

Employing a platform-independent approach in determining the translations during a two-step conversion requires 2(n) translations. In the same example of a 4 systems-federation, the number of translations will be 8. This method is not only more efficient and more productive in terms of cost and memory consumptions, but the safety integrity of safety critical data transmitted between systems increases with less translations, i.e. with 2-step method. Candidate

approaches will be evaluated based on which method is used or is addressed (suggested to use) for resolution of representational differences.

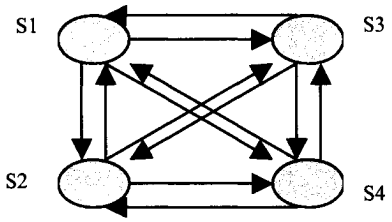

Figure 2.16- n(n-1) Translation

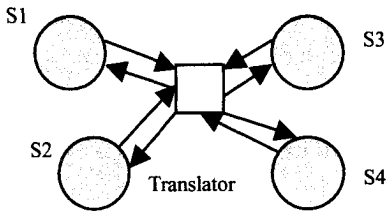

Figure 2.17- 2(n) Translation

Safe Federation Extensibility Capability: This capability refers both to computer languages used in the systems and to extensibility of the federation in terms of additions to the federation.

- A program is considered extensible if enhancements can be made to an existing component or data structure without adversely impacting the dependent entities. Dependent entities are those components that interact with original entity. A federation is safe-extensible when enhancements can be made to an existing component or data structure without compromising (or affecting) the designed-in safeguards in the dependent entities.
- A federation is considered extensible when additional systems can be added to the federation and changes can be made to information and interfaces among systems without adversely affecting interoperation of the original system federation. A federation is safe-extensible when additional systems can be added to the federations and changes can be made to information and interface among systems without adversely affecting the designed-in safeguards in the interoperations of the original system federation.

This safe federation extensibility provides the foundation for federation reuse (new upgrades, etc.) without concerns regarding safety critical data being impacted.

Candidate analysis approaches are compared to determine the level of support provided for creating an extensible system safely. If an approach meets one of the safe-extensibility support requirements, then it is considered partial support. If it meets neither support requirement, or meets one or both criteria with affecting or compromising safety, it is considered to provide no safe federation extensibility capability.

2.8 CURRENT APPROACHES TO ANALYZING SYSTEM SAFETY / INTEROPERABILITY

2.8.1 MIL-STD-882D APPROACH

MIL-STD-882D requires the System Safety Program (SSP) to establish a System Safety Working Group (SSWG) (USN, 2000). The SSWG is chaired by Principle For Safety (PFS) who is designated to oversee the entire phases of SSP, and to report the residual risk in the system to the stakeholder. The working group has a responsibility to define its interfacing IPTs and to establish an integrated working relationship with the rest of Systems Engineering teams.

In a typical single system level safety engineering (SSE) process, the safety engineer's or SSWG's interactions are concurrent with the rest of the development team. This SSE is not conducted in isolation, but parallel and to some extent integrated with the system engineering activities. Figure 2.18 shows the typical software safety engineering interactions / interfaces.

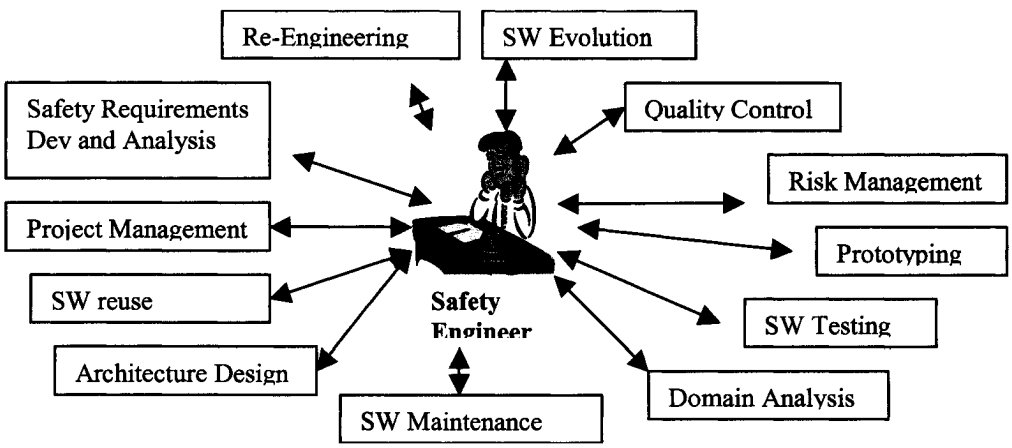

Figure 2.18- System Safety Working Group Interactions/ Interfaces

System Safety Process

MIL-STD-882 serves as guidance document for U.S. Navy System Safety Programs. Therefore it does not specify a process for system safety analyses. Each contractor or design agent can use its own process within the guidelines and the requirements of MIL-STD-882.

This general process, mentioned in paragraph 2.3.5, was developed in early 1970's to guide the new safety engineers in identifying and mitigating hazardous conditions in the system. Since the safety engineering effort concentrated on the single system with known input, known output and clear functionalities, this process was able to ensure an acceptable level of safety for DoD's war-fighting

systems. However, neither MIL-STD-882 nor other early founders foresaw the coming of system of systems architecture and battle-group-wide combat operations. Therefore the process and safety activities required for development of DoD's war-fighting systems did not include interoperation with other systems based on holistic systemic view.

Hazard Risk Assessment

MIL STD-882D provides guidance to assessing system risks. Two qualitative and quantitative factors play a major role in determining the risk- one is severity and the other is probability of the mishap occurrence.

Severity - Mishap severity categories are defined to provide a qualitative measure of the most reasonable credible mishap resulting from personnel error, environmental conditions, design inadequacies, procedural deficiencies, or system, subsystem, or component failure or malfunction. Suggested mishap severity categories are shown in Table 2.6 (USN, 2000).

Table 2.6- Suggested Mishap Severity Categories

Description	Cat	Environmental, Safety, and Health Result Criteria
Catastrophic	I	Could result in death, permanent total disability, loss exceeding $1M, or irreversible severe environmental damage that violates law or regulation.
Critical	II	Could result in permanent partial disability, injuries or occupational illness that may result in hospitalization of at least three personnel, loss exceeding $200K but less than $1M, or reversible environmental damage causing a violation of law or regulation.
Marginal	III	Could result in injury or occupational illness resulting in one or more lost work days(s), loss exceeding $10K but less than $200K, or mitigatable environmental damage without violation of law or regulation where restoration activities can be accomplished.
Negligible	IV	Could result in injury or illness not resulting in a lost work day, loss exceeding $2K but less than $10K, or minimal environmental damage not violating law or regulation.

These mishap severity categories provide guidance to a wide variety of programs. However, every program based on its size, complexity level, and management judgment, can make adjustments to this table. The mutual

understanding and agreement between the Program Manager and the Design Agent is required before the definitions are applied. Other risk assessment techniques may be used.

Probability - Mishap probability is the probability that a mishap will occur during the planned life expectancy of the system. It can be described in terms of potential occurrences per unit of time, events, population, items, or activity. A sample qualitative mishap probability levels are shown in Table 2.7 (USN, 2000).

Table 2.7- Suggested Mishap Probability Levels

Descriptio n*	Level	Specific Individual Item	Fleet or Inventory**
Frequent	A	Likely to occur often in the life of an item, with a probability of occurrence greater than 10^{-1} in that life.	Continuously experienced.
Probable	B	Will occur several times in the life of an item, with a probability of occurrence less than 10^{-1} but greater than 10^{-2} in that life.	Will occur frequently.
Occasional	C	Likely to occur some time in the life of an item, with a probability of occurrence less than 10^{-2} but greater than 10^{-3} in that life.	Will occur several times.
Remote	D	Unlikely but possible to occur in the life of an item, with a probability of occurrence less than 10^{-3} but greater than 10^{-6} in that life.	Unlikely, but can reasonably be expected to occur.
Improbable	E	So unlikely, it can be assumed occurrence may not be experienced, with a probability of occurrence less than 10^{-6} in that life.	Unlikely to occur, but possible.

*Definitions of descriptive words may have to be modified based on quantity of items involved.

**The expected size of the fleet or inventory should be defined prior to accomplishing an assessment of the system.

Risk classification based on mishap severity and mishap probability can be performed by using a mishap risk assessment matrix. This assessment allows one to assign a mishap risk assessment value to a hazard based on its mishap severity and its mishap probability. This value is then used to rank different hazards as to their associated mishap risks.

Evaluation of Safe interoperability Analysis Approach
In this section, the MIL-STD-882D system safety approach is evaluated against the criteria we established in paragraph 2.7. The result of comparison is summarized in Table 2.8.

Table 2.8- Evaluation of MIL-STD-882D Approach

Evaluation Criteria	MIL-STD-882D	
SoS-based	No	This standard was intended and designed for single-system safety program. The system of systems operational environment is not addressed by analysis/ tasks requirements.
Address of Model Correlation	Partial	Model correlation for entities at component level is addressed, but not at system level.
Knowledge / safety requirements of remote operations (interfaces)	Yes	Interface operations at system level are analyzed via SHA. Safety Requirements are addressed.
Feedback (Validation & Verification)	Partial	Validation and verification of messages transmitted to and from the single system is analyzed via SHA, but at single-system level only.
Translation methodology	No	Lack of system of systems approach precludes the need for discussion of translation methodology.
Safe Federation Extensibility Capability	No	Only changes to a single program are addressed via PHA and SHA and SSHA.

As shown in Table 2.8, the MIL-STD-882D's approach in analyzing safety interoperability requirements is not adequate. Since the standard does not cover the SoS architectural environment, the system problems with safety impact are not addressed as being part of the analyses. MIL-STD-882D is "partially" responsive, because most of safety critical issues can be mitigated thru at the single-system level, but interoperability issues with safety impact cannot not be addressed by the use of this approach.

2.8.2 USS NIMITZ APPROACH

As explained in paragraph 2.5, USS NIMITZ (CVN 68) Ship Self Defense Combat System Safety Program launched in 2002, for the first time, an initial step toward analyzing a combat system (i.e. system of systems) for safety. Since the requirement for a comprehensive system safety effort came near deployment, there was not much time to implement a well-thought-of system safety program. A Preliminary Hazard Analysis (PHA) was done to identify the system of systems hazards, their causal factors, and their relationships to the other systems in the federation (NAVSEA, 2002). No System Hazard Analysis (SHA), or Safety Assessment Report (SAR) on the Combat system level was performed. The risk of combat system was not assessed due to lack of methodology for assessment. Instead, the risk associated with each hazard in the Combat System Element (CSE) element was given to the same hazard that posed a safety risk at the overall combat system.

Evaluation of Safe interoperability Analysis Approach

Table 2.9 shows the evaluation of USS NMITZ System Safety Analysis approach against the evaluation criteria.

Table 2.9- Evaluation of USS NIMIZ Approach

Evaluation Criteria	USS NIMITZ	
SoS-based	Yes	The combat system safety program was based on system of systems operational environment. This was the first CSSP in its kind to view the combat system as one large system and conduct a "limited" system safety analyses.
Address of Model Correlation	Partial	Model correlation for entities at CSE level is addressed. No analysis or verification tests was made as to its accurate implementations of resolutions due to lack of safety analyses such as SHA at the CS level.
Knowledge / safety requirements of remote operations (interfaces)	Partial	Although system level interfaces with other systems were defined as to their existence and to their safety criticality, no analysis were done to verify that safety requirements have been implemented successfully and that all failure modes have been controlled. No SHA was done for CS due to program scheduling issues. Safety requirements were not traced to safety critical interfaces/functions. Additionally, the safety requirement identification was limited to each subsystem, i.e. within the scope of single system.

Feedback (Validation & Verification)	Partial	Although safety critical messages going across the interfaces were identified during the CSE SSP, no analyses or verification was made at CS level due to Program Scheduling issues. This verification would have been documented in SHA report, but no SHA was performed for USS NIMITZ CSSP.
Translation methodology	Partial	The traditional method is 2-step, that is 2(n) methodology, which is a safe method for message conversion. But no verification performed to verify the safe interoperation of safety critical functions.
Safe Federation Extensibility Capability	Partial	Changes to software or to the configuration can be made, but in the absence of a system of systems engineering methodology, each CSE will have to use the traditional SE to make changes.

Based on the results above, it is clear that this approach is inadequate to meet the needs of a SoS level safety program.

2.9 CHAPTER SUMMARY

In this section, the current literature was reviewed for its relevance to this research. Since the topic being researched has basis on three categories of literature, that is, Systems of Systems environment, Interoperability, and System Safety, all three categories were reviewed for their relevance. The system safety works included software safety review.

As part of this review, the gap in the literature was identified. In addition, the safety interoperability issues were identified for later use in analysis of AEGIS Ballistic Missile Defense 3.0 Program.

The safety interoperability concept was introduced and the issues having safety implications were addressed. In paragraph 2.7, the criteria for safe interoperability capability is developed and provided. The criteria will be used to evaluate the current and the proposed approach to analyzing the safe interoperability in the Navy's complex combat systems.

In paragraph 2.8 the current approaches to analyzing the interoperability issues with safety implications were addressed. There were two approaches that have been used to date—one is MIL-STD-882 and the other approach used for USS NIMITZ. The latter approach was an 'ad hoc' modification and/or deviation from the first approach. These two approaches were evaluated against the criteria in paragraph 2.7 to demonstrate their adequacy in analyzing the safety critical

interoperability issues in combat systems. As stated in each case, the approaches did not fully meet the criteria established in paragraph 2.7.

CHAPTER III

SYSTEM SAFETY INTEROPERABILITY FRAMEWORK (SSIF)

"The defense that a hazard was not foreseen is not available to those who do not use expertise appropriate to their profession."

-Justice Jackson, 1953

3.1 INTRODUCTION

As noted in Chapter I, the purpose of this research is to develop a framework to assist in studying the interoperability issues of complex systems of systems from system safety standpoint. The objective of Systems Safety Engineering is to, "... ensure that safety *consistent with mission requirements* is designed into systems..." (USN, 2000). Mission requirements, for the Navy's complex systems of systems, are to interoperate and perform joint tasks operations. Joint tasks operation means the capability of one system to employ the services of another system. These services represent the behavior of real-world entities modeled by the owner-system. And since these systems were developed independently, in different times, with different requirements, by different manufacturer, they will most certainly be modeling those system behaviors in different ways. In Chapter II, we discussed the impediments to interoperability, and we stated that the modeling differences were on top of the list, and some of these differences had associated safety impact.

In addition, system safety issues in interoperability were identified in Chapter II. These issues, although part of performance domain, have safety implication. The failure of a safety critical function is of great concern for a safety engineer, especially in the new environment such as system of systems where the stakes are higher, and the lines of boundaries are ambiguous, and the system problems are fluid and more complex.

3.2 SYSTEM SAFETY- A DIMENSION OF INTEROPERABILITY

As stated in Chapter II, interoperability, so far, has not been viewed as having any connection to safety. It has been viewed primarily as a performance issue. Even some safety engineers share that opinion. The idea of safety as a dimension of interoperability is puzzling to many. For many, system safety is synonymous with occupational safety or personnel safety only, and no interoperability issue seems to fit that concept. Advancing system thinking in system safety engineering by

categorizing system safety, as a dimension of interoperability is one of the significant by-products and contributions of this research.

The following is the list of various dimensions of interoperability that have emerged in engineering, and have been "communicated" to the engineering world.

- Performance interoperability is the ability of the systems to perform system operations, exchange information, and achieve the ultimate mission requirement in an effective manner. As a part of this dimension, the interoperability of database systems is a major player since the systems will be using available database information for operations.

- Training interoperability refers to being able to train across multiple systems, perhaps across different platforms and, within the training program, being able to jointly perform tasks.

- Security interoperability is another aspect of interoperability that has been getting increased attention. Its primary focus is access control and integrity of data classification. Do those who have access or who want access to other systems have a "need to know" requirement? Is the integrity of information passed among systems monitored? Are the information we pass to other systems to conduct certain types of actions secure? Encryption capability is used almost always to enhance the security interoperability of the systems.

- Testing interoperability refer to coordinated testing effort to verify that the hardware, middleware, operating environments and the software are able to interoperate. In this area, the architectural integration requirements are tested and the focus is on the hardware and environmental issues as opposed to software functional operations. This testing is sometimes called "conformance" testing and it is referred to checking the conformance of the entity to a profile and its normal operation and to detect errors. A profile is defined by the standard used in each layer of the OSI model. The process and methods of conformance testing must be in accordance with ISO 099646 (ISO, 1987). This testing overlaps with the performance testing in that it tests for capability of the system, the behavior and interconnections.

- Communication/ Information interoperability refers to the ability of different communications systems to be used for the interoperability of different systems. Message traffic in most network-based systems is based on message services that enable the exchange of information between applications (Benkhellat, Siebert, and Thomasse, 1993). In DoD, Link 11, 16 are typical media for systems to interoperate (from communication perspective.) Among the different physical communication systems, there is multilingual conversion systems that different services or allies in battle

group can use to communicate and interoperate together using their native language.

- The forgotten dimension is system safety. System safety interoperability refers to the ability of the Federation system to perform its mission requirement without posing a safety risk to personnel, equipment (includes weaponry, the ship, other assets) and the environment.

Figure 3.1 shows the many dimensions of interoperability.

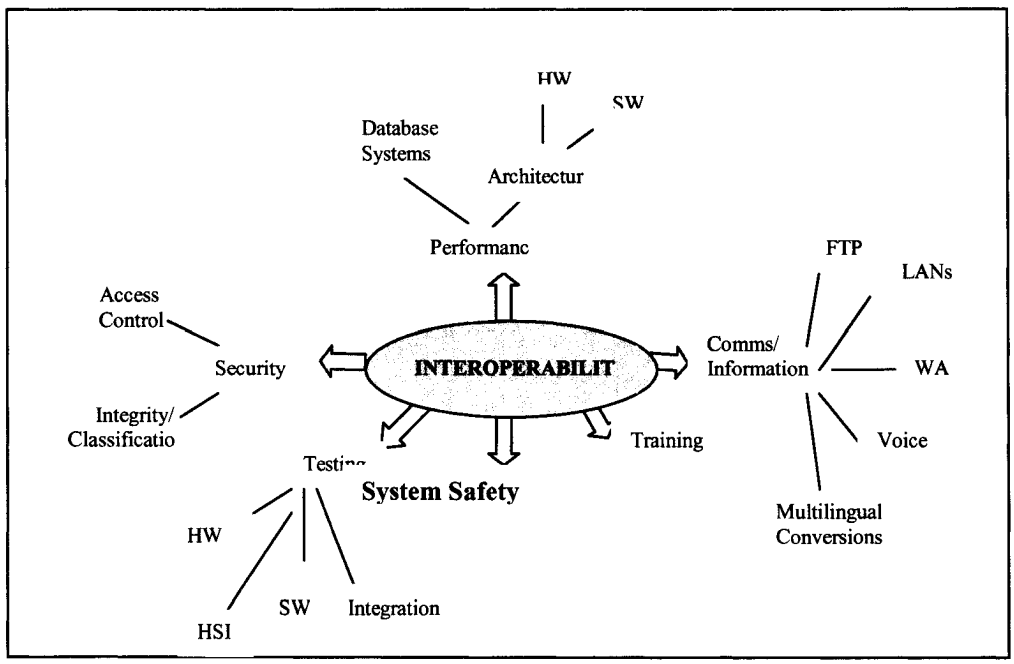

Figure 3.1- System Safety- A Dimension of Interoperability

3.3 SAFETY CHALLENGES ASSOCIATED WITH SOS

As stated in Chapter II, currently all complex Navy war-fighting systems are required to have a system safety program (SSP) by which the hazards of the system are identified, eliminated and/or controlled to an acceptable level.

In compliance with NAVSEAINST 8020.6D (NAVSEA, 1997), the SSE is an integrated component of SE (see Figure 3.2). The integration means that not only that the conduct of SSE tasks will be concurrent with the program development, but safety will be an integral part of the "criteria" on which the effectiveness of a functional element of the program is measured during the decision makings, development, coding and testing and analysis. Safety critical requirements and functions are identified and disseminated throughout development team to be used

for safe implementation of design. Safety engineer interfaces with the design and test team to ensure the safety properties are implemented and designed into the system.

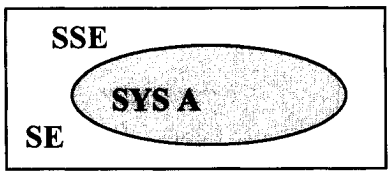

Figure 3.2- SSE an Integral Part of SE

In a tactical System of Systems (SoS) environment, however, the mechanism by which the safety SSP can be engineered and produced does not exist. The challenges facing system safety are:

- **Lack of Standards.** These are Specifications that can govern the SoS level Safety Program.
- **Lack of SoSSE-** SoS Safety Engineering (SoSSE) is in its conceptual level of creation at this writing. The current SSE is inadequate to address the non-linearity of today's developments. It is also inadequate in addressing the safety features for interoperations between systems through design.
- **Lack of SoSSE Methodology-** No methodology exists at this writing to support the engineering and design of safety critical features for safety critical systems. The methodology, when developed, must be in concert and harmony with SoSE. The methodology must address interoperability requirements, and must be flexible and adaptive.
- **Lack of adaptable/software-focused analysis tools-** FTA does not seem to respond adequately to identification of software safety issues in SoS level. No analysis methods exist to analyze the safety interoperability of the system. Also no analysis tool exists to analyze the safety robustness of the SoS. The manual analysis on these areas is resource-intensive and error-prone.
- **Lack of overall SoS safety interoperability framework to be used in guiding the SoSSE program for the Navy's combat systems.**

As a part of this research, a framework (called System Safety Interoperability Framework) within which an analysis of SoS could be conducted is developed and provided. The System Safety Interoperability Framework (SSIF) will be the focus of discussion in this chapter.

3.4 SSIF CHARACTERIZATION

System Safety Interoperability Framework (SSIF) is designed to serve as a high-level philosophical and conceptual framework by which a system safety program for system of systems environment is established. The framework intends to be flexible in adapting to specific requirements of the meta-system, and also adaptable to the fluid nature of system of systems engineering challenges.

Characterization Attributes

The SSIF characterization attributes are those attributes that make up (or characterize) the essence of SSIF. These are:

- SoS- System of Systems
- SoSE- System of Systems Engineering
- SoSSE- System of Systems Safety Engineering
- SCD- Safety Critical Data

As a part of the research methodology, SSIF was reviewed and validated by three subject matter experts. The validation and approval of SSIF by the three subject-matter experts are in Appendix B of this dissertation along with the expert-election process and a short biography of each expert. The validation of the experts attest to the potentiality it demonstrates for its soundness in being implemented in analysis of AEGIS BMD program.

Spiral Characterization Attributes

Although, the attributes are listed and explained separately, one after the other, in reality and at work, they are overlapping and spiral in nature; they are not separate, linear or isolated. To further explain this, SCD analysis will be in majority of SoSSE's activities such as, robust analysis, safety interoperability analysis, safety integrity analysis, etc. Moreover, the activities within SoSSE will be spiral as well; that means the safety integrity analysis will overlap with safety interoperability analysis and that will overlap with software analysis, etc. SoSE will include SoSSE verification and validation, etc. The SoS existence is the precursor to all other attributes, and so on.

Figure 3.3 displays this spiral relationship.

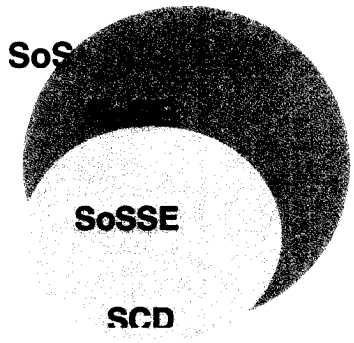

Figure 3.3- Spiral Characterization Attributes

SSIF is presented in Figure 3.4. In the following sections, each SSIF characterization attribute will be discussed in detail.

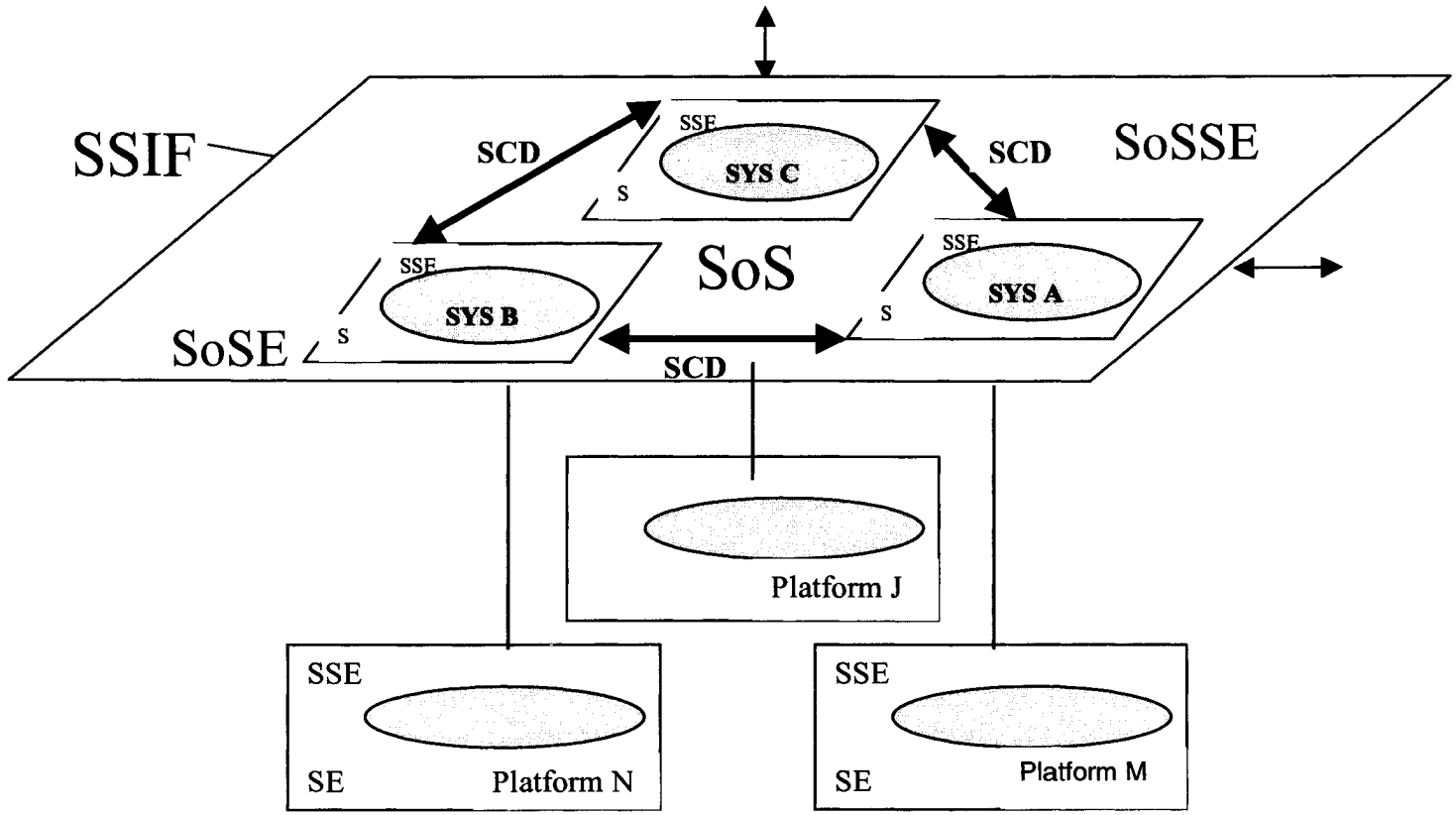

Figure 3.4- System Safety Interoperability Framework

SE: Systems Engineering
SCD: Safety Critical Data
SSE: System Safety Engineering

SoS: System of Systems
SoSE: System of Systems Engineering
SoSSE: System of Systems Safety Engineering

3.4.1 SoS

It is important to remember that all discussions on SoS are limited to U.S. Navy's combat systems. When two or more independent, autonomous complex systems are integrated and interconnected as one larger system to perform a higher level of mission requirement, the result is a system of systems. In Figure 3.5 below:

System A is an autonomous, complex system, e.g. a sensor system, installed in platform M. This system was developed under traditional SE and had been analyzed for safety via SSE, and therefore the hazards in the systems had been identified, eliminated or controlled during development.

System B is another complex, autonomous system that was developed through the traditional SE under separate requirements, and is now being modified or reconfigured to integrate with System A. System B is installed in platform N. This system has also been evaluated for safety under SSE at the time it was developed. It is fair to assume that system-wide hazards have been resolved through the SSE.

System C, another complex autonomous system, e.g. a missile system, has been also developed through its own SE process and had been evaluated for safety through SSE. This system is either modified and/or reconfigured to integrate and interact with System A and B to accomplish a different outcome (than it was designed for). System C is installed in platform J.

These three systems are interconnected and integrated and will be required to perform joint execution of tasks. In fact these three systems are now one larger, more complex system, called a system of systems, a meta-system, a federation (of systems), or in military terms, a combat system. The integration, interrelation and interconnection capability is accomplished by SoS Engineering (SoSE), and that is the topic of discussion in the next section.

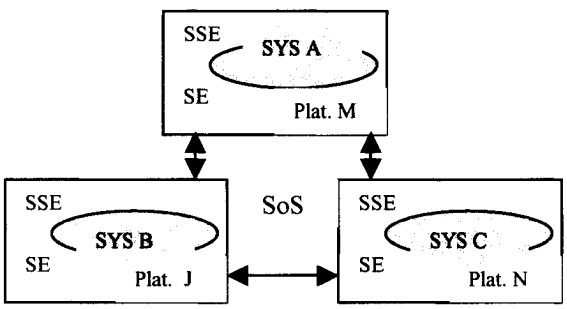

Figure 3.5- Complex System of Systems

3.4.2 SoSE

In SoS operational environment, multiple legacy, new or hybrid complex systems are integrated and configured to create a larger, more complex system whose operational outcome is greater than any individual system's output. The SoSE is an evolution (more than an extension) of traditional Systems Engineering (SE).

For this discussion, we use the same example as above, three systems, A, B and C. Each systems has been developed under individual SE program development, with no interoperability requirement when it was designed, and may have diverse operating environment (OE), computer languages, mission requirements, etc. Each system may be deployed in different geographical location, on a different platform, and under different military service, etc. each system has had an individual system safety program and therefore each individual system hazards have been either eliminated or controlled in its level. Figure 3.6 shows the three heterogeneous systems that will be integrated to create a system of systems.

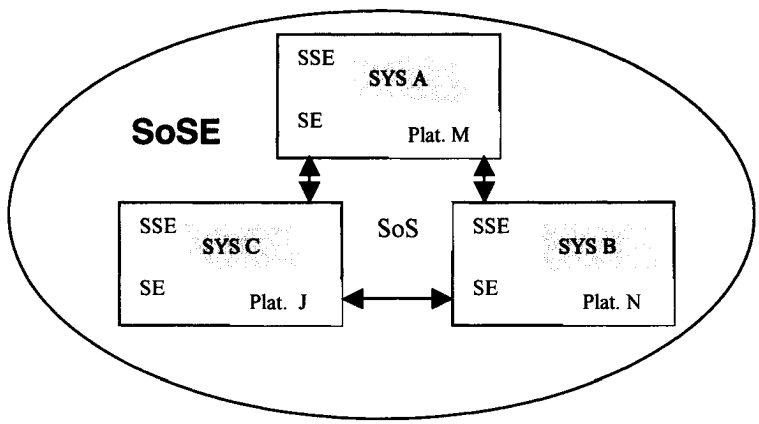

Figure 3.6- System of Systems Engineered via SoSE

The means by which the system of systems can be designed and produced is SoS Engineering, and specifically, SoSE methodology.

SoSE Methodology

SoSE methodology development is in the embryonic stages (Keating, Sousa-Posa, and Mun, 2003). The methodology to engineer systems of systems must be flexible and adaptable. This methodology must resolve interoperability impediments that are hinders to effective performance and safe operation. SoSE must have an ability to oversee the effective progression of teams from one phase to another without adversely affecting the proper implementation of system of systems requirements and system of systems safety requirements. This methodology will support verification and validation of safety interoperability requirements. The verification shall include the "no-path" as well as "path"

testing. The "no-path" testing refers to "negative" safety requirements within the engagement sequence.

Ability to Expand

Another highlight of the methodology is that it allows for the expansion of the Federation. Its adaptive nature and flexibility allows for extensibility capability. SoSSE will be an integral part of SoSE, in that its methodology to conduct the SoS safety program will be "mirrored" to SoSE methodology in respect to phases of development and the analyses of artifacts. The SoSE methodology must be responsive to the dynamic nature of SoS environment. Keating suggests (Keating, Sousa-Posa, and Mun, 2003) that the meta system requirements must be developed from a holistic systems perspective, and they must consider the wide array of organizational, managerial, and contextual constraints (not just technological constraints) influencing the development of the complex system.

Guided by System Principles

Keating and Sousa-Posa developed a set of guiding principles to direct and assist the development of SoSE methodology (Keating, Sousa-Posa, and Mun, 2003). Chapter II, paragraph 2.1.1 lists these system principles. Of these systems principles, the following are most essential for the safe interoperability of the system:

- System control
- System context
- Boundary Establishment
- System Outcome Achievement
- Complex System transformation
- Self Organization
- Iteration

Again, it is important to remember that SoS safety engineering will be relying in part on SoSE for the designing safety into the meta-system.

Reliability/ System Failures

A part of SoSE is to develop criteria for reliability threshold. The system reliability is directly linked to system failures. System failures are typically caused by inadequacy of requirements, mismatches, misinformation, incompatibility, system crashes, etc. About a third of reliability failures are also safety failures (Alborzi, 2002). Figure 3.7 shows this linkage.

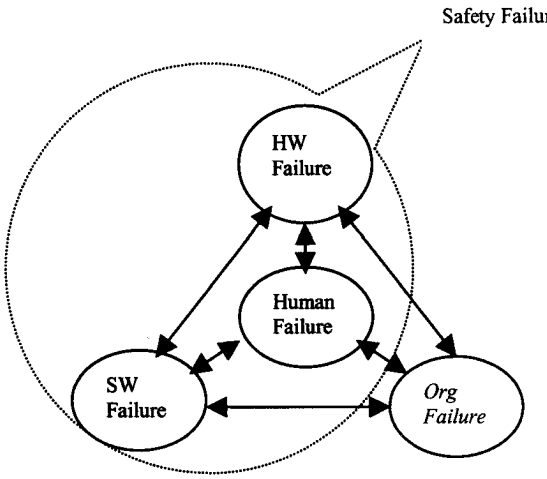

Figure 3.7- Reliability Failures and Safety Failures Overlap

SoSE focuses on software reliability. It attempts to create redundancy or by other means to increase the software's fault tolerance. Although, this may help a little in discovering the coding errors and other errors associated with implementation of requirements, it does little to prevent safety faults. This overlap presented above must act as a catalyst in SoSE to create a culture change, one that truly integrates safety into its design, where safety can become an indivisible property of the system. It must be understood in SoSE community that merely increasing reliability, does not necessarily increases the safety of the system.

Many engineers believe that safety and reliability are one and the same. The Figure 3.4.e shows that, indeed, there is an overlap, but the two are not the same. Reliability is focused on preventing component failures and increasing system's availability. Safety is focused on accident prevention. Components failures can occur without an accident occurring, and an accident can occur without a component failure.

There are usually three reasons why accidents occur even though the reliability of the system is high and acceptable (Leveson, 1999):

1. The software correctly implemented the requirements, but requirements specify behavior that is not safe from a system perspective (safety requirements analysis would be able to catch this item.)

2. The requirements do not specify some particular behavior that is required for safety (new safety requirements derived from requirements analysis and added to the system specification would mitigate this issue.)

3. The software has unintended (and unsafe) behavior beyond what is specified in requirements. (Only code analysis by a safety engineer can discover this problem.)

Modeling Differences
Some of the incompatibilities, mismatches, and misinformation are part of modeling differences that must be resolved in SoSE. The interoperability impediments need to be resolved during the engineering effort. Recall that modeling differences were on top of the list when interoperability impediments were discussed.

Interoperability Resolution Methods
To enable heterogeneous systems to interoperate and perform joint execution of tasks, these modeling issues need to be resolved before system safety issues are analyzed. The SoSE must employ a method or methods to resolve these modeling differences. Two methods were discussed in Chapter II:

1. The Object-Oriented Method for Interoperability. This analysis uses OOMI as the foundation for the study of system safety interoperability of complex systems of systems.

2. The Holistic Framework for Software Engineering (HFSE) was also discussed in chapter II. This framework allows software engineers to trace dependencies, to document decision-making process, and all the important information, and make it available to all development team for a coherent, consist use of data. The strength of HFSE is that it is integrated with Quality Function Deployment, a requirement-based methodology to ensure the accuracy, designation (for example designation of safety) and tracing of requirements.

The OOMI and HFSE will be the foundation on which SSIF will be employed. If used, these two approaches for resolving interoperability issues will be a part of System of Systems Engineering (SoSE) methodology. As an alternative, any means of resolving interoperability issues associated with modeling issues and associated with lack of coherent framework for software development, can be implemented. SSIF assumes the interoperability capability is a property of the design.

As an integral part of SoSE (see Figure 3.8) the SoS Safety Engineering will need to be conducted simultaneously and in concert with SoSE to resolve the interoperability issues and failures that pose safety risk. The next section will encompass the integrated SoSSE.

3.4.3 SoSSE

The means by which a meta-system can be analyzed for safety is through a SoS Safety Engineering (SoSSE) program. This program will focus on the safety

issues involving the meta system. The hazards at the subsystem level have been resolved at the system level within the SSE. The hazards that exist and originate at any of the subsystems and can contribute to the overall risk of the meta system, that is, can contribute to a mishap at the meta system level will be identified, analyzed, and corrected via SoSSE, which is an integrated element of SoSE. Figure 3.8 shows this relationship. The "integration" into SoSE requires teaming with other development groups of systems in the federation. This teaming is conceptualized by an integrated system safety working group. This will be discussed in the following paragraph.

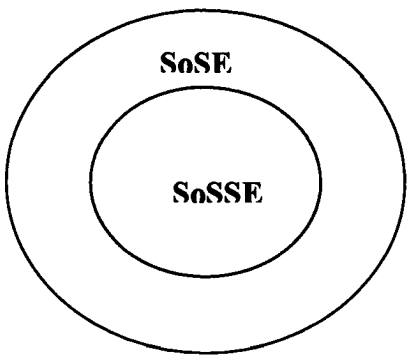

Figure 3.8- Integrated SoS Safety Engineering

SoSSE Methodology

SoS safety engineering methodology is a mechanism and an engineering mean by which the safe performance of mission requirements is achieved. As stated earlier, no formal, structured, and disciplined methodology exists for SoSSE at this writing.

The following are the essential 'ingredients' of development and execution of an effective safety methodology for SoSE:

1. It must be system-based.

2. It must be based on system safety principles.

3. It must be responsive (flexible) to the dynamic and fluid nature of SoS development.

4. Safety Requirements must be derived, analyzed, and verified from holistic system perspective

5. It must be developed on system principles. System principles essential for development of a SoSE methodology are included, but not limited, to:

a. Unity of System purpose. This means the subsystems must structure the safety activities, objectives and approach with maximizing safety of the combat system (not individual system).
b. Iterative (Gibson, 1991). This means that designing and deploying and transforming complex SoS are iterative processes that evolve, as more information becomes known.
c. System Context (Keating, Sousa-Posa, and Mun, 2003). This principle refers to the set of relevant circumstances, conditions, and patterns that both constrain and enable the development of system solutions, operation, deployment and transformation.
d. Compatibility (Keating, Sousa-Posa, and Mun, 2003). This principle refers to compatibility of objectives for the redesign and the approach taken for those objectives.
e. Dynamic Stability (Keating, Sousa-Posa, and Mun, 2003). The ability of system to respond *positively* to the unknown turbulent events in the operational environment is one of the safety objectives for the safe performance of the SoS.

6. It must contain processes that are not too prescriptive, but allows for creative thinking.

7. It must be in harmony with SoSE methodology. They must compliment each other not be in conflict with each other.

Integrated System Safety Working Group

Integrated System Safety Working Group (ISSWG) consists of representative from system engineering, development, testing effort plus users, stakeholders of each system of Federation, and safety boards. The Principals for Safety and safety engineers from all participating systems will be permanent members. The Principal for Safety (PFS) of the overall SoS chairs this working group. Figure 3.9 shows the interactions of ISSWG with the various development teams.

The responsibility of each person is to report any concerns that seem to be inhibitive to work's objectives. Each PFS for subsystems is responsible to identify and report any hazards, causal factors contributing to any hazard that may have safety impact on the overall system of systems performance to the SoS PFS. The analysis and risk assessment of the identified SoS safety hazards are the responsibility of the SoS PFS or designee.

The sponsors, stakeholders or their representatives have a responsibility of ensuring that funds are available for SoS level safety program and that issues concerning schedules, resources or safety boards are promptly taken care of.

As a part of SoSE, the modeling problems between systems will be resolved, and the resolution will need to have safety engineer's involvement to ensure the safe implementation of the design. OOMI and HFSE are two methods for resolving the heterogeneity of the systems that pose safety risk.

It is the responsibility of ISSWG members to discuss and finalize the SoS level hazards, potential mishaps, causal factors, and all interoperability issues. It is the responsibility of this working group to finalize the list of tasking tailored to the needs of the overall system and prepare a safety schedule to accomplish these tasks.

The combat system PFS will be the oversight authority to assess the quality and effectiveness of the SoS SSP and review testing reports for verification of safety requirements. The final assessment of residual risk of the overall combat system will be reported by the combat system PFS to the sponsoring agent.

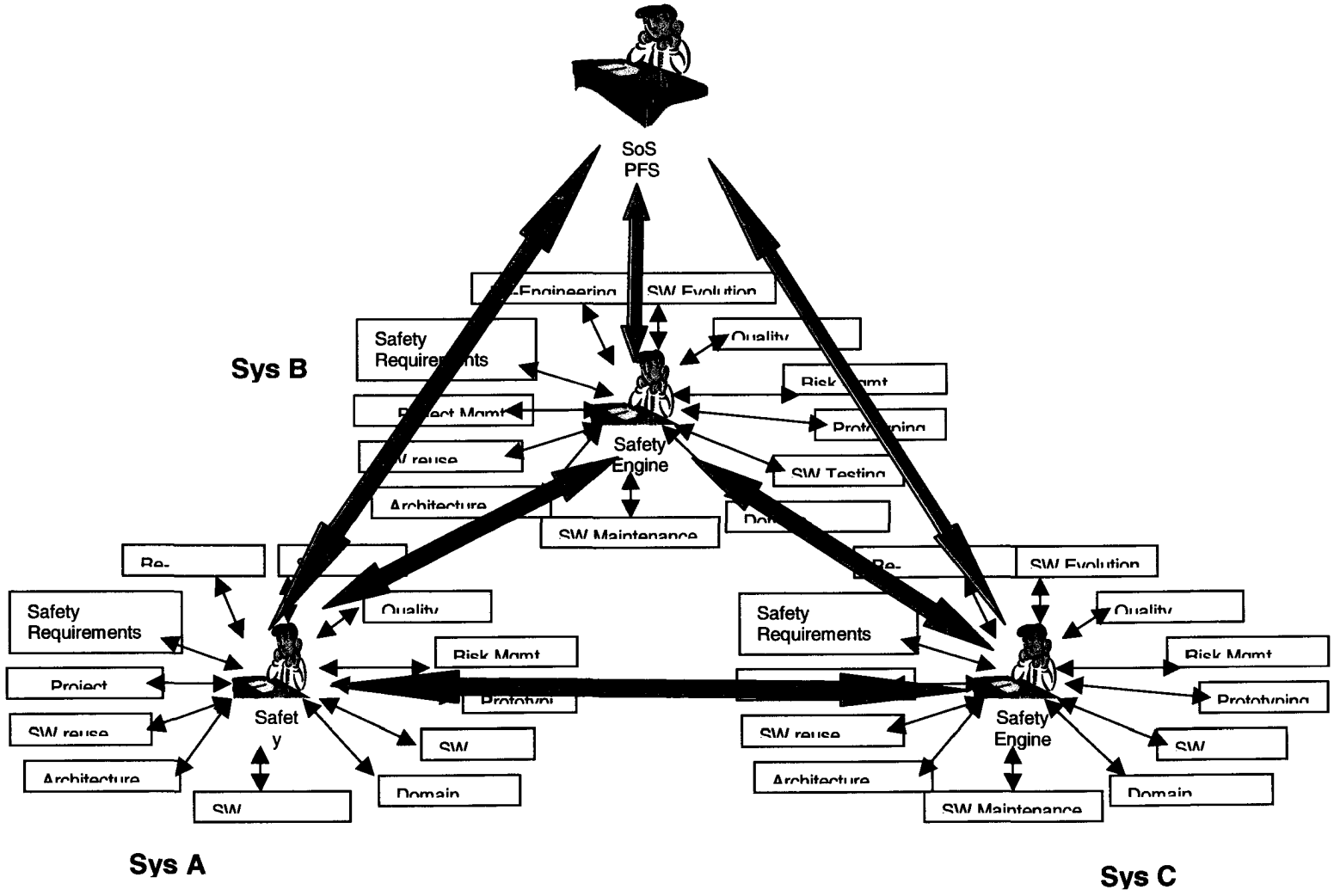

Figure 3.9- The Integrated System Safety Working Group Interactions

Tasks / Activities

As a part of SoSSE, the following activities or tasks will be essential for conducting an effective SoS safety program. Note that these activities are not "systematic" in nature, that is, step one does not always precede the second and the second does not always precede the third, etc. These activities are iterative, overlapping and "circular" in nature. The safety engineering for SoS, in most part, mimics the SoSE in that it is flexible and fluid in nature. It is able to change, to conform, to adapt, and to use synergy form other activities and products.

Program Plan- a comprehensive program plan must be written to encompass activities, schedules, processes and methodologies for analysis of all involved subsystems. The plan will be signed by design agent and by stakeholders. The roles and responsibilities of the SoS safety program team must be delineated in a clear manner. This program plan is in addition to the individual systems program plan. This plan focuses on the combat system safety program.

Safety Requirements (SR) Analyses (includes Interoperability Requirements)- Each system will perform its own requirements analysis. The Requirements that are applicable to the overall mission, at the SoS level, and are safety related will be identified, flagged and discussed at ISSWG. The safety engineers will do analysis for requirements' completeness, soundness, testability, applicability and safety impact. The interoperability requirements are analyzed for their source-destination aspects, their definitiveness of starting action and conclusion action on the part of systems involved, the content of information being exchanged. Each requirement may be aggregated to smaller requirements, creating "nodes". This definition is vital on the safety testing events. Figure 3.10 shows this definition.

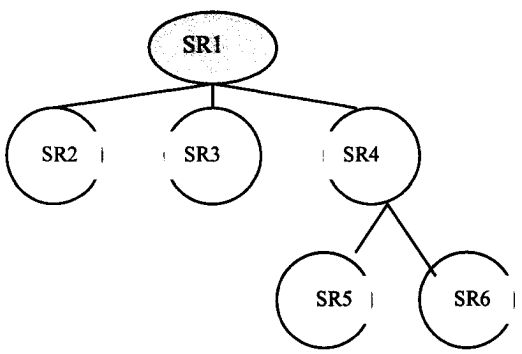

Figure 3.10- The Node Aggregation of SoS Safety Requirements

For example:

SR1: The overall System state shall be Tactical, Training or Offline.

SR2: The offline state shall be when the system is unable to support operations, including power off, operational software loading, system wide maintenance, or system failure condition.

SR3: The Tactical states shall be the normal powered on state, including the ability to support mission operations, processor failure, and replay.

SR4: The Training state shall be when the team training is planned to be performed across platforms.

SR5: The Training status shall be displayed at all consoles.

SR6: The Training tracks shall be designated with a "T" for easy distinction.

The requirements analysis process must include interrelation (as graphed above), development of a matrix, discrepancy/ incompleteness resolution, and development of definition (that is a final list of safety requirements). The designations of these safety requirements in the overall development matrix must be visible to all SoSE teams.

Hazards Identification and Analysis- Hazards are identified for their impact on SoS. The casual factors are identified and reported to development team. SoS Hazard Analysis (SoSHA) will include the discussion of what are the interface requirements, interfaces and interactions between systems, what types of Safety critical data (SCD) are passed and acted upon by systems, etc. it will also include recommendation or statements for controls, safety kernel, safety features, for mitigations.

Identification and Analysis of Safety Critical Functions (SCF)- These are the software that computes the safety critical data (SCD). In U.S. Navy systems, any function that controls either directly or indirectly the pre-arming, arming, launching, firing, or detonation of a weapon system is a safety critical function. The "indirectly" applies to the functions that are used (contributors) for engagements, for example, the identification friend or foe. Other SCFs are Break Engage, Hold Fire, and Cease Fire.

HMI or Operational & Support Analysis
Human Machine Interface (HMI) analysis is performed for operator-related errors as the result of interaction of operator with the software. In other words, these are software-driven errors made by the operator. Interface issues such as displays are also analyzed.

Safe Interoperability Analysis- This analysis consists of identifying what are interoperability issues that pose safety risk, finding the root causes for the problems, and ensuring the resolution of the those problems. As a part of SoSE, the modeling problems between systems will be resolved, and the resolution will need to have safety engineer's involvement to ensure the safe implementation of the design. OOMI and HFSE are two methods for resolving the heterogeneity of the systems that pose safety implications.

SSIF is not contingent upon the use of these methods but is contingent upon the use of a method that resolves the modeling difference and requirements dependency issue by the development team within the SoSE. The problems are typically format difference, representation differences, validity differences, etc as stated in Chapter II.

The interoperability analysis differs slightly from SHA-System Hazard Analysis, even if it were to be expanded to apply to SoS. The difference is that SHA's primary focus is to ensure the integrity of the interface and the successful transmit-receipt of messages. Interoperability. The interoperability analysis focuses not only on the exchange of information, but also the integrity of interoperation of functions or tasks being requested.

Safety Robustness Analysis

This analysis is not typically done in system safety primarily because the "robustness" is thought of as a design-for-performance feature. But unsafe situation arise from some "collision" of interactions that failed a test, or lacked information or input that it needed to have, even in non-safety critical portions of the software; note that non-safety critical software are interconnected with safety critical software.

There are three items (Leveson, 1999) that makeup the criteria for Safety Robustness:

1. Every state must have a behavior (transition) defined for every possible input.

2. The logical OR of the conditions on every transition out of any state must form a tautology.

3. Every state must have a software behavior (transition) defined in case there is no input for a given period of time (a timeout).

The above criteria ensures that if there is a trigger condition for a state to handle inputs within a range, there will also be a clearly defined transition to handle data that is out of range. The third criterion ensures that there will be a requirement for a time-out that specifies what to do in case no input is received.

Safety Threads- As a part of interoperability analysis, the safety critical 'threads' are identified and analyzed for safety of operation. For example, when a system within the federation is requesting the other system to break an engagement because it just received a valid "friendly" ID, the thread of interactions need to be analyzed and verified for the safe interoperation. Figure 3.11 shows this concept.

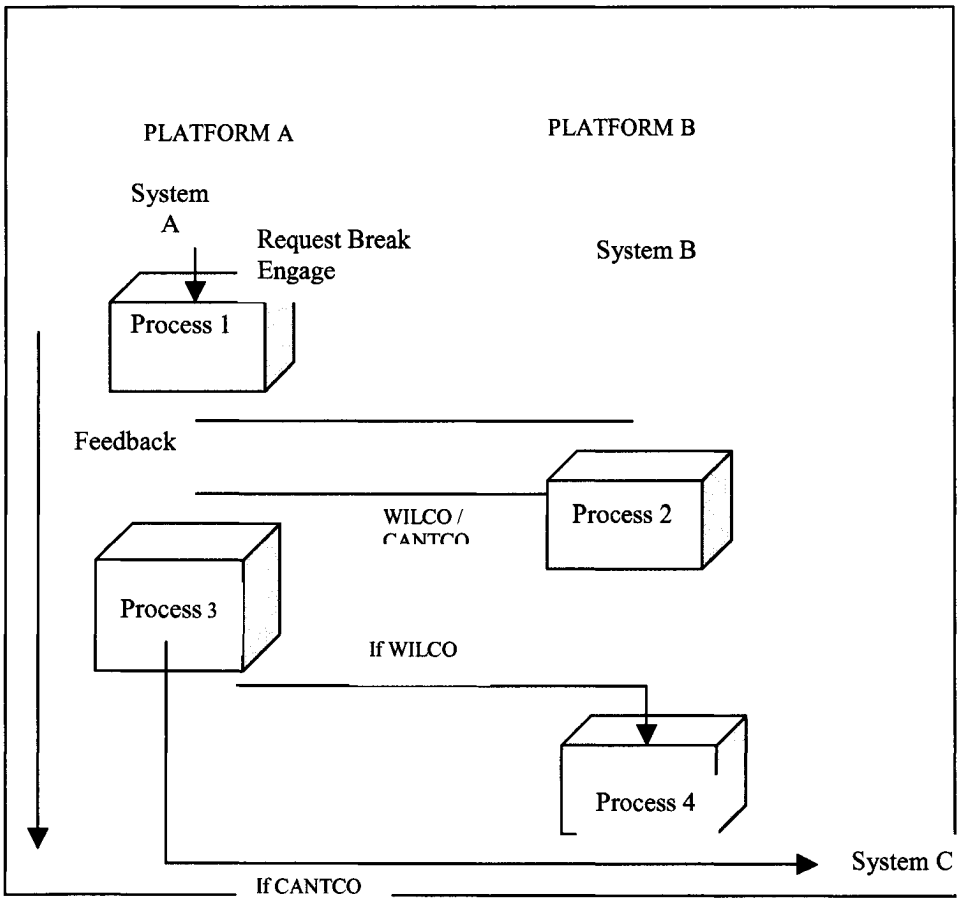

Figure 3.11- An Example of Safety Thread Analysis

Note that processing 3 refers to the processing after processing 2, that is, the sequence of events, not a "name" of a processing. To clarify this even more, once processing 2 is completed and a feedback is received, based of whether the feedback is WILCO or CANCO, a processing will be initiated that either will go to system B or system C, but it will be one action taken based on the feedback, and that is why we call it processing 3.

Verification and Validation- The SoS safety program will invest a great deal of effort on verification and validation of SoS safety requirements, and verification of implementation of design in the source code. The validation includes the transmission of messages from another systems and its reception in the receiving

system, its understandability and its acceptance. This requirement also ensures that the "command" or "response" needed from the receiving system is validated to be the proper response at the proper time. This implies "decision-making" capability by the receiving system to comply with a request or wait for further information. This type of validation is essential for the safety of the system's operation.

The safety testing at SoS must be an integrated, concerted effort with the SoS Testing team. The envisioned safety testing is conceptualized in Figure 3.12.

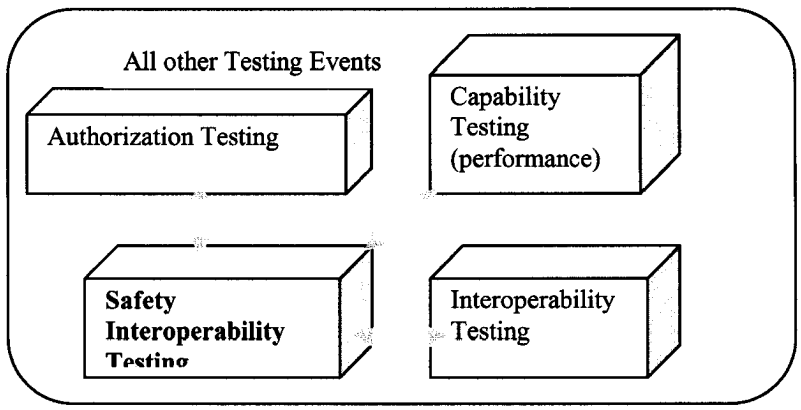

Figure 3.12- Safety Interoperability Testing for SoS

3.4.4 SCD

Safety Critical Data (SCD) is the next topic to discuss, as it is one of SSIF characterization attributes. SCD are computed during run time by the SCFs and are analyzed as a part of the SoSSE. SCD exists in all safety critical applications and systems, such as missile systems, command and control systems, explosive systems, and nuclear systems. It is the robust design and thorough safety analysis of SCFs and SCD interactions that identifies, eliminates or controls the risk in the system.

The crucial role of analyzing the SCD in SoS, is evident when we remember the recent past's accidents of complex systems. The Three Mile Island incident, the Challenger disaster, the Russia's Chernobyl nuclear power plant accident, the Shuttle Colombia's tragedy, all are evidence for the need to conduct a *preventive* system safety program that is system-based, logical, and with depth and breath that it requires.

Safety Integrity of Data

In software safety the *safetyness* of SCD is a great concern for engineers. No matter how well the logic of software is written, if the data that will populate that logic is faulty, then the results will be faulty. Faulty or erroneous data can cause a system to operate unsafely, and may lead to a catastrophic accident. Stalhane says, "the software system always communicates with its environment through data values, so dangerous events are...connected to incorrect data values" (Herrmann, 1999).

Some of the ways to monitor the safety integrity of data is to:

- Ensure that intended data is correctly accessed (keywords are *intended* and *correctly*.)
- Ensure that SCD is not corrupted
- Ensure the validity of data
- Checking for out-of-bound parameters
- There is no unused data
- Ensure that there is no memory conflicts
- Ensure error detection/ alerts reporting for SCD
- Identify Safety critical interfaces
- Ensure the integrity of tactical data and training data
- Ensure no self-modifying code, especially in safety critical functions
- Check for unused or dead code (and remove them from the source code)
- Ensure that "Exceptions" are processed correctly

Hazard

Hazard is a condition of a system that, together with other conditions in the environment of the system will lead inevitably to an accident (Leveson, 1999). The SCD contains any combination of those conditions stated above. The identification of hazards in the system starts with studying the SCD of the system. The main objective is to 'discover' the hazards before the system is ready to operate.

Since we are speaking of "data" and software, the hazards can only be within the scope of its system's design, not associated merely with software. For example, the hazard, *Launch on non-hostile*, is associated with a function called, *Launch Procedure*, with, say, 100 source lines of code. The 100 lines of code, in themselves, are neither safe nor unsafe, but when the code is executed, and directs the control of a launcher to launch a missile at a "friendly", we have a catastrophe. So the hazard of *Launch on non-hostile*, associated with the *Launch Procedure* is in existence, but will not create an accident unless the code (SCD) executes and interacts with the system, and based on the wrong ID (SCD), it fires at a friendly unit.

Role of SCD in Accidents

Accidents often occur not only by one contributor or causal factor, but by a combination of interactions between human, system and their environment. In

complex system of systems, all components and subsystems including human and the environment on which the meta system is operating are interacting directly or indirectly. Safety can be looked at as a quality of the system, and so it is controlled by set of controls or enforced by a set of constraints. Accidents occur when one or more of these constraints are violated when components are interacting. The violation can be a lack of constraint or a constraint that was over a threshold in real world. In complex system of systems, software is the controller and so it is software that enforces the constraints since it controls the interaction of the components. Software, then, is a contributor to accidents by not enforcing the constraints (safety critical features) or by bypassing a constraint and so violating it. Accidents, in complex systems, occur because a safety critical function of the system failed and the failure had an impact on the rest of the interactions and lead to an accident. As a part of software analysis, the safety engineer must identify safety critical functions (SCF) and safety critical data (variables, messages, etc.) to the developers for accurate implementation of design. Early involvement of engineers enables them to influence the design so the right constraints and controls are planted in the code for safe execution later.

From discussion above, it is clear that the integrity, accuracy, and validity of SCD in interoperability (the interactions of components of systems) are vital to the safe operation of the system of systems, and this, in turn, testifies to the crucial role it plays in interoperability and the reason behind being an attribute of the SSIF characterization.

3.5 EVALUATION OF SSIF AGAINST SSI CRITERIA

3.5.1 Initial Validation of SSIF

As a part of evaluation of its viability to be used for analysis of Aegis BMD program, the SSIF was presented for review and validation to three subject matter experts. The three experts have many years of engineering and system safety engineering experience and have served as Principal for Safety for U.S. Government for several of Navy's complex war-fighting systems. A short biography of each expert can be viewed in Appendix B.

The criteria used by subject matter experts are as follows:

- Applicability to SoS operational environment
- Capture of interoperation capability
- Lessons-Learned (from Gulf war, Iraq War, and near-miss accidents)
- Professional Experience

Three subject matter experts developed the criteria and the list is the compiled criteria. For example expert one may have had 2 of the four criteria above, and expert 2 may have had 3 out of four, and expert three may have had another combination of three out of four. The author asked for the criteria from each

expert and each list was combined without redundancy (that is if two experts has lessons learned, it was stated once, as it is shown above). The list above shows the compilation of experts' criteria used for validation of SSIF.

The detail on this initial validation is provided in Appendix B. Appendix B also includes the process used to choose the three experts and the transcripts of the approval.

3.5.2 Evaluation of SSIF

Now that we described SSIF and explained how it is characterized and what the characterization attribute encompasses, it needs to be evaluated against the system safety interoperability criteria that we developed in paragraph 2.7. Table 3.1 shows the results of the evaluation.

A: SoS-based
B: Address of Model Correlation
C: Knowledge /Safety Requirements of Remote Operations (interfaces)
D: Feedback (validation &verification) Capability
E: Translation Methodology (messages, representation of entities, etc.)
F: Safe Federation Extensibility Capability

Table 3.1- Evaluation of SSIF

Evaluation Criteria	**SSIF**	
SoS-based	Yes	This approach is designed for SoS operational environment.
Address of Model Correlation	Yes	This approach is founded on Object-Oriented Method for Interoperability (OOMI). This method enables the resolution of modeling differences and enables interoperability of joint tasks. Some of these modeling conflicts are safety critical, and the resolution of these items will be necessary for the success of SSIF.
Knowledge / safety requirements of remote operations (interfaces)	Yes	This approach has the requirements analysis as one of the core tasks and it is done within the SoSSE. Not only the knowledge of the remote interfaces is within the framework, but also the analysis of the interfaces to facilitate the safe operation of tasks exchanged between systems.
Feedback (Validation & Verification)	Yes	Verification and validation refers to the ability of the system to give feedback to the transmitting system that a message (and the type of message) is received. It also refers to the ability of the system to validate a message for further action.

		SSIF is designed to allow this capability in its SoS safety program for safe operations.
Translation methodology	Yes	As a core entity of the system of systems, translator is used to resolve the many kinds heterogeneity of systems. One method to use is the OOMI using 2(n) architecture. SSIF is intended to be used on basis of OOMI for resolution of modeling differences used in messages, languages, representations, etc.
Safe Federation Extensibility Capability	Yes	SSIF is capable to be extended. The adaptive design of SSIF is to allow for change in configuration, by adding new systems. The adaptive and flexible SoSSE allows for such extensibility.

Based on this evaluation and the validation of three experts, SSIF will be used to evaluate the safety interoperability issues in AEGIS Ballistic Missile Defense baseline 3.0. After the implementation, the SSIF will be adjusted, if needed, to represent a framework that can be used to analyze the real-world system of systems from safety standpoint.

3.6 CHAPTER SUMMARY

In this chapter, the many dimensions of interoperability were discussed and the argument on why safety is a dimension of interoperability was clearly explained. The dimensions of interoperability were summarized as:

- Performance
- Training
- Testing
- Communication / Information
- Security
- System Safety

We also learned about the challenges facing system safety in relation to the Navy's system of systems environment. The challenges were:

- Lack of Standards
- Lack of SoSSE
- Lack of SoSSE methodology
- Lack of adaptable/software-focused analysis tools
- Lack of overall SoS Safety Interoperability Framework to be used in guiding the SoSSE program.

In this chapter, we presented a System of Systems Safety Interoperability Framework that can be used to frame the problems associated with interoperability of safety critical data. Three subject-matter experts using their own developed criteria initially validated this framework. Then, we looked at the system safety interoperability framework (SSIF) and learned how it is characterized. The characterization attributes discussed were SoS, SoSE, SoSSE, and SCD. Each characterization attribute was explained in detail. The highlights of each attribute are:

- **SOS:** Integration of two or more legacy, heterogeneous complex systems into one larger, more complex system.

- **SoSE:** Methodology, ability to expand, guided by system principles, reliability/system failures, modeling differences, interoperability resolution methods.

- **SOSSE:** Integrated w/SoSE, integrated SSWG, tasks/activities, program plan, safety requirements analysis, hazard identification/analysis, safe interoperability analysis, safety robustness analysis, safety threads, verification and validation.

- **SCD:** Hazard, role of SCD in Accidents

At last, we evaluated the SSIF against the criteria we had developed in Chapter II for analysis of safe interoperability issues of complex combat systems, and it clearly met all the criteria.

In the next chapter, the SSIF will be used to analyze a real-world complex system of systems.

CHAPTER IV

ANALYSIS OF AEGIS BALLISTIC MISSLE DEFENSE (BMD) 3.0 USING SSIF- *A USE CASE*

The [FAA] administrator was interviewed for a documentary film on the [Paris DC-10] accident. He was asked how he could still consider the infamous baggage door safe, given the door failure proven in the Paris accident and the precursor accident in Windsor, Ontario. The Administrator replied—and not facetiously either—"of course it is safe, we certified it."

-C.O. Miller
A Comparison of Military and Civilian Approaches to Aviation Safety

4.1 EXECUTIVE SUMMARY

In Chapter I, it was stated that this research intends to answer two questions:

- What is system safety interoperability?
- How can safety interoperability be analyzed for a complex combat system?

The previous chapters have been successful in answering the first question. In Chapter I, we defined system safety interoperability as,

"A capability encompassing many of the safety issues relative to integration, compatibility and interface that are impinging upon the effectiveness with which independent, heterogeneous and/or homogeneous systems, components or elements, including human factor, may safely interact"(Alborzi, 2004).

We also provided one hypothetical example in Chapter I, paragraph 1.6, "Safety Interoperability", and another example in Chapter II, paragraph 2.6.2, "Pre-Mature Launch" to demonstrate the concept of system safety interoperability. In this chapter a system safety interoperability statement will be provided that further establishes what system safety interoperability is in the Navy's combat systems.

The second question is how can SSI be analyzed. The two existing approaches for analyzing a complex system of systems in respect to safety interoperability were presented and evaluated (see Chapter III) against the criteria that were derived as the result of literature review, and the results were noted in Chapter III. The SSIF approach, that is, analyzing a complex system of systems for safe

interoperability issues by using a framework was developed, validated by three subject-matter experts, and evaluated against the same criteria. The SSIF met all the criteria, and therefore it will be used in this chapter to analyze the AEGIS Ballistic Missile Defense 3.0 program.

NOTE: As stated in chapter I, "Assumptions/ Limitations", no classified information will be used in this analysis, and all "sensitive" information will either not be used or will be denoted with "X" for security reasons. In addition, this chapter will not be released to the public without the written approval of the U.S. government. All attempts are made to keep this analysis high-level without an effect on the content and integrity of the analysis.

4.2 AEGIS BMD BLOCK 04 SYSTEM DESCRIPTION

Mission Statement

The primary AEGIS BMD mission statements are (BMD, 2004):

•To Deliver an enduring, operationally effective and supportable Ballistic Missile Defense capability in Aegis Cruisers and Destroyers, for the defense of the U.S., our deployed forces, allies.

•To increase the effectiveness of the greater Ballistic Missile Defense System (BMDS) by both providing and gaining synergy from other BMDS elements

Background

Missile Defense Agency (MDA) is planning the development and fielding of the missile defense in two-year blocks, (BMD, 2004) with each block more capable than previous one. The first block- the Block 04 consists of:

Aegis BMD 3.0E – provides Aegis Destroyers with Long Range Surveillance & Track (LRS&T) capability. BMD 3.0E is authorized to provide LRS&T support to Ground-based Missile Defense (GMD) systems.

Aegis BMD 3.0 – provides Aegis Cruisers with the capability to serve as a test bed for SM-3 Flight Test Missions and if ordered to do so, provide an Emergency Deployment capability. BMD 3.0 is planned for installation in FY-05 and will be authorized for test engagements of short and medium range ballistic missiles and LRS&T support to GMD. Pre-production SM-3 BLK I missiles will be used. Aegis BMD 3.0 is the program to be used in this research as a use case for analysis of safety interoperability capability based on SSIF.

Aegis BMD 3.1 – provides Aegis BMD engagement capability and is currently planned for deployment in FY-06. Aegis BMD 3.1 will be certified for BMD mission requirements and will provide a limited self-defense capability. SM-3 BLK IA production missiles with an X year shelf life will be employed. Full

implementation of Aegis BMD 3.1 will consist of up to X SM-3 BLK IA missiles on X Aegis Cruisers and X LRS&T equipped Aegis Destroyers.

Figure 4.1 shows the mission concept (Aegis, 2004).

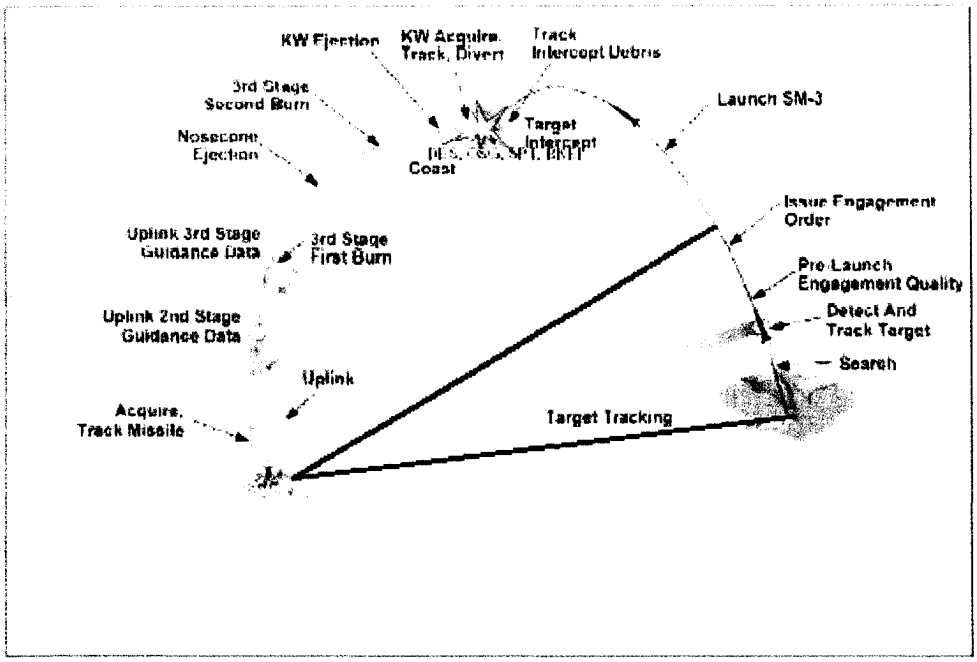

Figure 4.1- AEGIS BMD 3.0 Engagement Mission

System Description

Aegis BMD (ABMD) is a system of three systems. ABMD system of systems is, in itself, a subsystem to the BMDS. The three systems forming a system of systems are:

- AEGIS Weapons System
- Vertical Launching System
- Standard Missile-3 System

The SoS or combat system for ABMD system is shown in Figure 4.2.

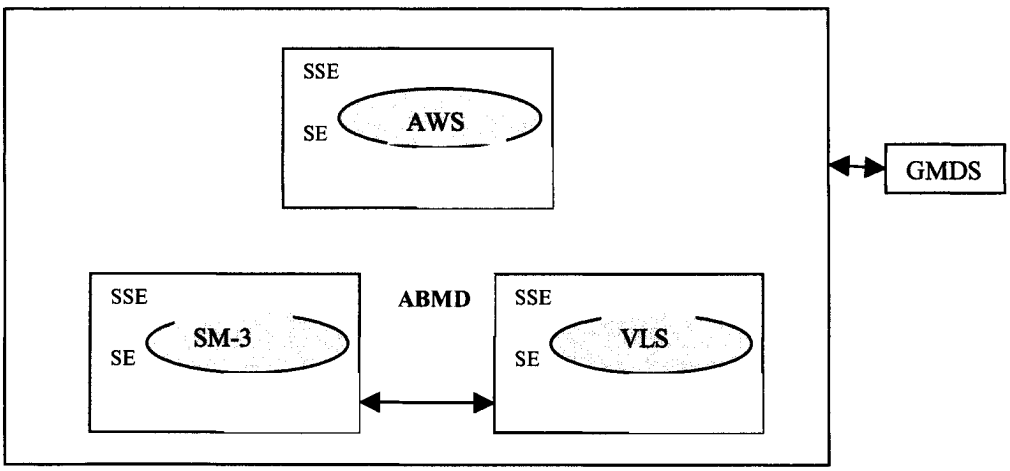

Figure 4.2- AEGIS BMD Combat System

4.2.1 Aegis Weapon System (AWS)

The AWS is a shipboard weapon system characterized by rapid reaction, continuous availability, high firepower, and environmental immunity.

The AWS is further decomposed to more complex subsystems, and it is shown inside the blue dotted lines, Figure 4.3. The AWS subsystems include:

- Weapon Control System (WCS)
- AN SPY-1/1B Radar system (SPY)
- Command and Decision System (CDS)
- Mission Planner (MP)
- Aegis Display System (ADS)
- Aegis Training Control System (ACTS)—Not operational for BMD missions
- Operational Readiness Test System (ORTS)

Functional Capabilities

AWS is a command and control system capable of detecting and engaging threats. Table 4.1 provides the functional capabilities of AWS. Safety Critical Functions is determined from this table.

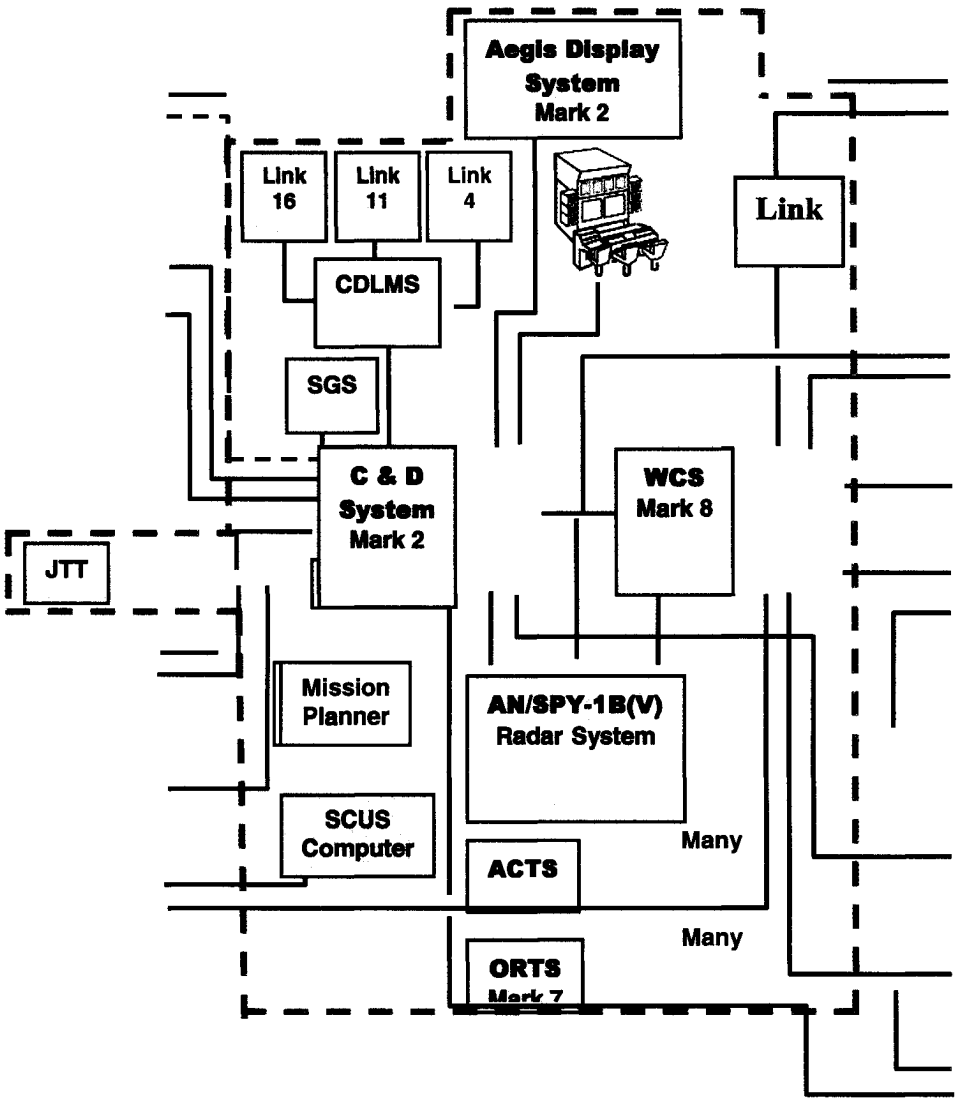

Figure 4.3- Aegis Weapon System

Table 4.1- Functional Capabilities of AWS

	AWS SUBSYSTEMS			
Functional capabilities	**MP**	**SPY-1**	**CDS**	**WCS**
PLAN	1-Launch/ defended areas entered 2- doctrine defined			
DETECT		1-Predefined search sectors available 2-search sectors developed by MP using intelligence 3- initial Surveillance and Tracking capability		
CONTROL			1- Automated Engagement capability 2- Exchange link 16 data 3- Receive TDDS data	
ENGAGE				1. Pre-launch and engagement logic, guidance laws, and offset guidance 2. Updated RF link 3. Discrimination/designation design 4. Air control - voice
ASSESS				Kill assessment

4.2.2 Vertical Launching System (VLS)

MK 41 VLS is a general purpose launching system capable of launching missiles for air, surface, and underwater engagements. As a part of the ship's total combat system, the VLS has the capability to load, stow, select and fire the missiles. Each missile is launched perpendicular to the ship's horizontal reference plane from a canister contained within a VLS launch module and located below deck. The electronically operated cell deck hatch, together with the module top surface, forms the weather deck and provides ballistic protection for the missiles (missile rounds) contained in the launcher.

VLS components include:

- Launch Control Computer Program (LCCP)
- Launch Sequencer (LSEQ)
- VLS GPs Interface (VGI)

VLS is further decomposed in Figure 4.4.

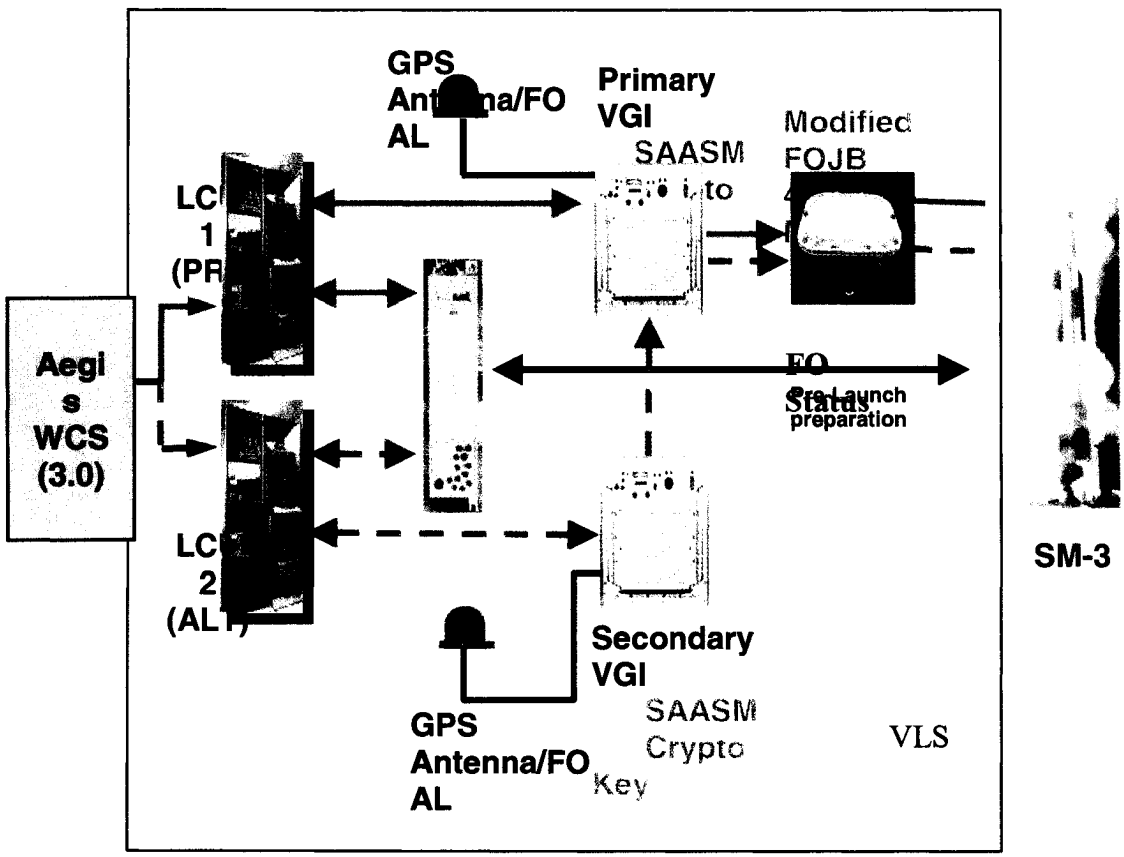

Figure 4.4- Vertical Launching System

Functional Capabilities

Table 4.2- Functional Capabilities of VLS

Functional Capabilities	COMPONENTS				
	LSEQ	VGI	LCCP		
	FUNCTION				
Launch SM-3	1. MSL interface control 2. Control motor Control Panel via command /response interface 3. Power supply interface control 4.Auto Test control	Receive GPS data	*-Prepare, launch and safe missiles* -Provides interface for communications between a Weapons Control System (WCS) and VLS, as well as VGI and LCU –Validates WCS messages and forwards VLS responses to WCS *-Provides communication with launcher equipment LSEQs*		
SAASM		Use militarized GPS receiver for improved tracking and anti-spoofing			
CRYPTO key Load		Use cryptographic keys for GPS processing 2. Generate GPS hot start data and GPS timing & transmit to the MSL.			

4.2.3 STANDARD Missile-3 (SM-3)

The SM-3 is guided through four phases of flight:

Boost: During the pre-launch period, batteries are activated, built-in tests (BIT) are performed, and the booster arms and ignites. The boost phase is the period from missile's first motion until booster electrical separation from the second-stage after approximately X seconds. Guidance provides booster nozzle commands, limits total lateral angular acceleration of the missile, and controls

missile dispersion to ensure missile beacon capture by AWS. Boost guidance is performed autonomously onboard the missile.

Midcourse endo-atmospheric (stage 2): The midcourse endo-atmospheric phase is the period from booster separation to third stage separation. Uplink AWS midcourse guidance missile body acceleration commands provide control fin commands to maintain missile stability and execute WCS steering commands. The second stage is guided by WCS to the next command point and is released from the third stage.

Midcourse exo-atmospheric (stage3): The midcourse exo-atmospheric phase is the period from third stage separation to KW separation. Guidance provides TSRM nozzle angle and roll attitude commands to maintain missile stability and guide the missile for minimum KW zero effort miss. Third stage guidance is effective only during TSRM burn.

Terminal phases: The terminal phase is the period from KW separation to intercept. KW guidance is autonomous once the KW IR sensor has acquired the target in its field of view. KW IR sensor target tracking data provides body axis divert commands to intercept.
SM-3 is further decomposed in Figure 4.5.

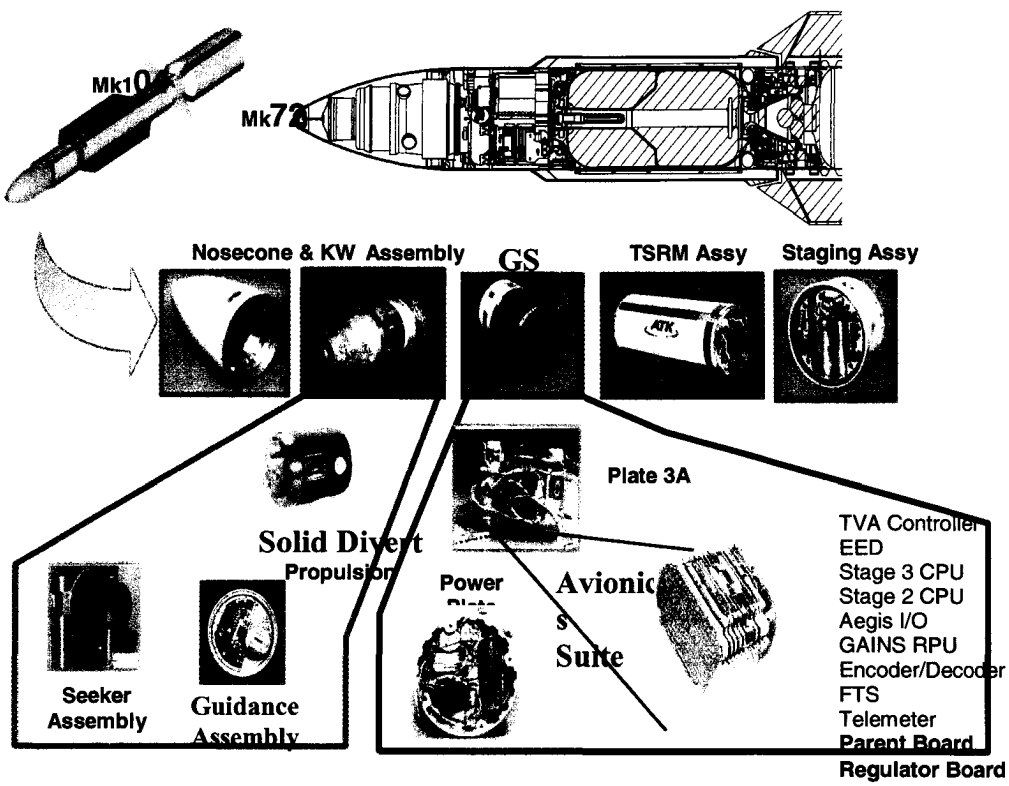

Figure 4.5- STANDARD Missle-3

Functional Capabilities

SM-3 system has capability to intercept air, surface, and underwater threats, including multi-mission launcher capability (SM-2, SM-2 Block IV, VLA, TH, and SM-3). Included in the above capability is the capability to load, stow, and select the missiles, and to perform Built-In-Test (BIT).

Table 4.3- Functional Capabilities of SM-3

Functional Capabilities	COMPONENTS			
	DTRM	TSRM	KW	Booster
	FUNCTIONS			
BIT	X	X	X	
Firing	1 Initialization message processing 2. MSL ready 3. Processing link data 4. Midcourse auto pilot guidance	1. Nosecone ejection, 2. In-flight alignment, 3. Initialization and ejection of the KW, 4. Third Stage operation. 5. Processing of uplinks and downlinks, 6. Control of DTRM separation, 7. TSRM burns 8. X gas generators	1.Target Tracking 2. Generate KW position 3. Aim-point selection 4. Target intercept Guidance 5. IR Target discrimination 6. IR Feature Fusion 7. Uplink/ downlink data requests	1.Battery Activation 2. Ignition

4.3 ANALYSIS

The basis for the analysis is the System Safety Interoperability Framework (SSIF). SSIF will be applied to this meta-system, also referred to as the combat system, for analysis of interoperability issues from system safety standpoint. As stated in Chapter III, these activities and the use of SSIF characterization attributes are spiral and non-linear in nature. The analysis approach is not purely systematic, but primarily systemic and iterative. However, some activities will have to happen before others. For example the requirements analyses will be performed before the safety integrity analysis. We will "iteratively" be involved in the domain of the following four attributes during the analysis. The first two

attributes are prerequisites for an effective and successful safety interoperability analysis, thus they are discussed first.

- One of the attributes of SSIF characterization is System of Systems (SoS). For ABMD 3.0, we identified the systems that are required to be integrated and interoperable in paragraph 4.2, and provided detail descriptions of each system's capabilities, and the role it plays in the SoS. More detail on identification and the safety criticality is explained in SoS attribute in paragraph 4.3.1.

- SoSE will be discussed to some extent. This discussion will include the interoperability "enablers" that are applied to the SoS, and have been discussed in previous chapters as methods to resolve interoperability conflicts.

- The SoSSE will be performed to analyze the ABMD. In this analysis, this effort will be 'abbreviated' and tailored since, 1) we are only evaluating the interoperation activities that have safety implications, 2) risk assessment is not the objective of this evaluation. As a part of the safety engineering and analysis, safety critical interfaces, functions, potential mishaps, hazards, will be identified and evaluated.

- The SCD will be discussed (as a part of SoSSE). SCD will be computed in the safety critical functions and will be related to identified hazards, interfaces, and potential undesirable events. Safety interoperability analysis will be done using the SCD.

The Approach Used to Apply SSIF to ABMD

Since interoperability is made possible by data being transmitted and used by systems in the federation, for a high-level mission requirement, the "integrity " and safety of that data is crucial for an effective interoperation. The Safety Critical Data (SCD) is data used, manipulated, and transformed by safety critical functions of the meta system. During the design, it is safety critical functions (SCF) that are identified, evaluated for "go" and "no-go" paths. Before that, it is the safety critical requirements that reveal the SCFs. The safety critical requirements are identified by overall system requirements analysis and by derived requirements based on whether or not the existing requirements meets the safety principles and precepts. For this task, the system specifications, especially Interface Design Specification (IDS) between the safety critical systems are reviewed. The interface criticality of the subsystems is determined by using "interface analysis".

Since the scope of this analysis does not include hardware related risk assessment (determining the probability and severity of a hardware failure), the current analysis tools such as Fault Tree Analysis (FTA) or Energy Trace Barrier Analysis (ETBA) will not be useful. Also, since this analysis is limited to safety implications of interoperability-related software, it is determined that current FTA and ETBA are not adequate to meet the needs of software analysis. To assess the overall software risk, however, an analysis tool(s) must be developed to respond to many different combinations of behavioral situations within a piece of safety critical software. This deficiency will be reflected in "Recommendations" section in Chapter IV.

In this analysis, we will look to see how SSIF can be used to analyze the safety interoperability issues. Therefore, 'abbreviated' version of an ideal SoSSE methodology will be used. Also, the analysis tool will be "interface analysis" that is shown in the analysis approach.

Figure 4.6 shows approach taken to apply the SSIF to ABMD for the purpose of evaluating safety interoperability capability.

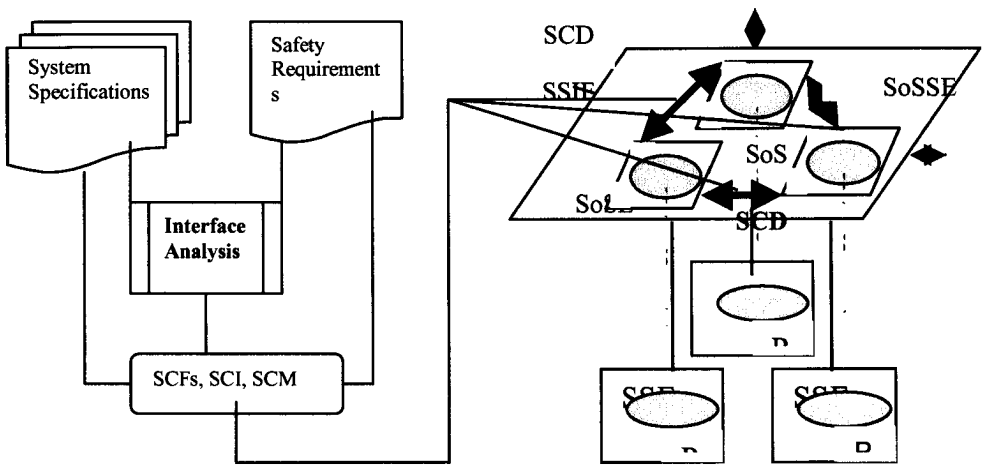

Figure 4.6- Approach for SSIF Application

4.3.1 ABMD System of Systems (SoS) Identification

Recall Figure 4.7. SoS is identified as the combat system consisting of three complex systems, AWS, VLS and SM-3. The combat system is in itself a subsystem to BMDS. It interfaces directly with Ground-based Midcourse Defense System (GMDS). GMDS system is also a subsystem to BMDS. The following graph shows the hierarchy of the systems of systems, and the ABMD SoS relationship to other systems.

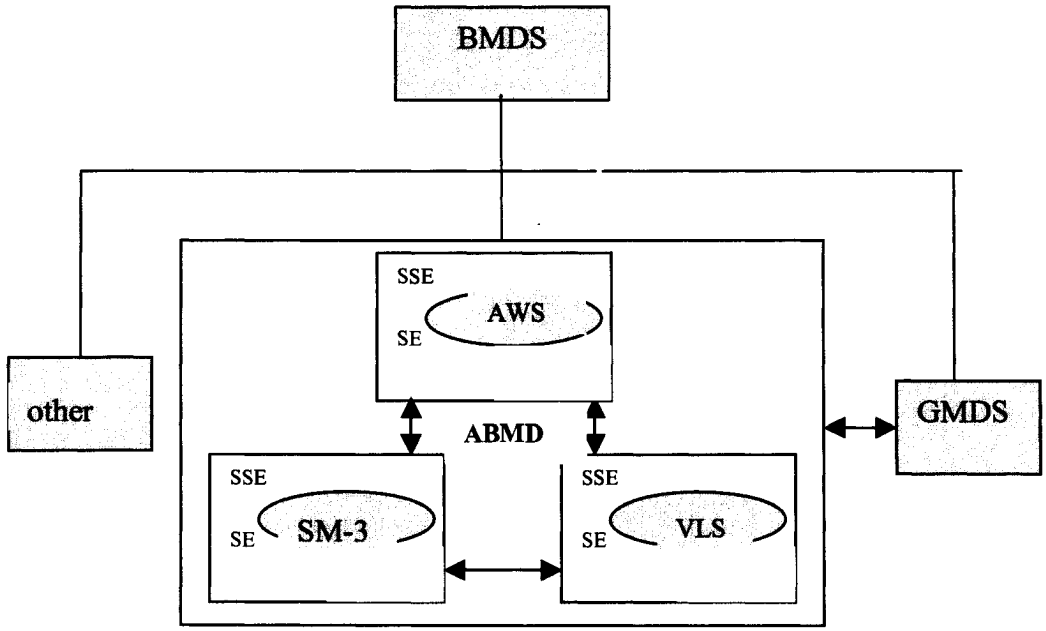

Figure 4.7- AEGIS BMD SoS

Findings: None. The SoS requirements of SSIF matched the ABMD SoS.

4.3.2 Aegis BMD System of Systems Engineering (SoSE)

ABMD SoSE is a mechanism by which the BMD SoS is designed, produced and deployed and /or transformed. The SoSE Integrated Product Team (IPT) consists of the following at the minimum:

- Requirements
- Design
- Code
- Testing
- **System Safety**
- Security
- Logistics/support
- Training
- Reliability/ Maintainability/ Availability

The Aegis BMD used a current single-system development process. There were modifications/ deviations when necessary to accommodate the new architecture.

Recall the SSIF's SoSE attributes of chapter III:

- Methodology and its characteristics,
- Ability to expand,

- Guided by system principles (System control, system context, boundary establishment, system outcome achievement, self-organization, iteration.)

The ABMD was analyzed using these attributes during system engineering effort. Figure 4.8 shows the SoSE methodology used for Aegis BMD. *(Note: The rest of the analysis will use this figure to evaluate the applicability of SSIF.)*

System control was clearly demonstrated by CDS component of AWS. System context was clearly defined by ABMD. This was demonstrated by modifying the configuration to "clean up" the context. In other words, the systems and capabilities and functions that did not match the system context (that is a system of systems environment for the purpose of engaging a ballistic missile) were removed, and so the system context was defined and documented in its true sense.

Iteration occurred during software engineering and unit testing (following the concept of build a little, test a little). Functionality or a computer program defect was fixed, tested, then fixed again (if needed) and tested again. Then at the end the system was tested as a whole.

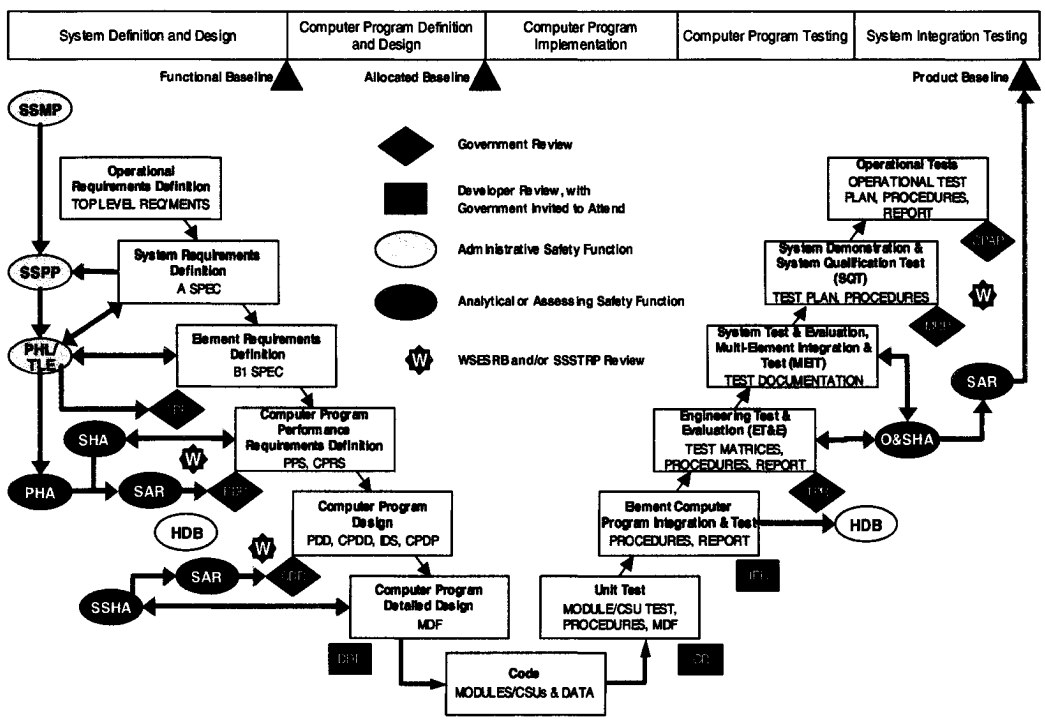

Figure 4.8- ABMD SE Methodology (Aegis, 2004)

Findings
The first 2 bullets above were checked successfully as the SSIF met the requirements for extensibility, methodology (shown above) and based on system principles. The system of systems engineering methodology envisioned in SSIF surpasses that of Aegis BMD SoSE methodology in that it allows for self-organization
It is the recommendation of this research that any methodology design chosen, must be founded on system-based attributes (Keating, Sousa-Posa, and Mun, 2003) *plus* system safety principles and precepts. Note that this is in addition to safety precepts embedded in SoS safety engineering.

System Safety Precepts- a 'partial' foundation for development of a SoSE methodology

a. Safety shall be designed into the system of systems and be reassessed if transformation is required.

b. There shall be positive measures to prevent exposure of personnel to the hazards of radiation, high voltage, toxic materials, excessive noise levels, and other potential personnel hazards created by system of systems.

c. There shall be positive measures to prevent the inadvertent firing of a weapon.

d. There shall be positive measures to prevent the chance of fratricide of weapons that could result in catastrophic mishap.

e. The system of systems shall provide for positive measures to minimize the danger to friendly forces from weapon firings.

f. There shall be positive measures to prevent harm to the environment.

4.3.3 ABMD System of Systems Safety Engineering (SoSSE)

SoSSE effort must be in parallel and in agreement with SoSE effort. The primary reason for this requirement is that SoSSE must be an integral part of SoSE if it is going to be effective in engineering safety into the system design via SoSE. SSIF was applied to ABMD parallel to system analysis activities.

Safety Criticality Criteria
The following is the list of criteria and definitions developed and used, to identify the safety critical or related subsystems or components for Aegis BMD SoS.

- Impact on the ship/weapons, assets (e.g. controlled aircraft, helicopter, satellites, shuttle, etc.), equipment
- Impact to environment
- Impact on the war-fighters (includes own forces, all friendly forces).

Safety critical subsystem or component is defined as an entity that directly or indirectly controls the release of ordnance or provides data for the launch sequence functions.

Safety related subsystem or component is defined as an entity that contributes or influences a safety critical function and whose failure contributes to the safety risk of the system.

The approach shown in Figure 4.6 enabled the effective identification of the safety critical subsystems/components, interfaces, and messages based on the criteria above. The identification of potential undesirable events leads the analyst to identifying the hazards that may be contributory factors. Since the scope of this research does not include risk assessment, the mitigating factors or controls need not be identified.

Safety Critical Subsystems/ Interface Identification
Safety Critical Interfaces:
The interface between a safety critical subsystem/component to another is safety critical interface. For ABMD SoS the safety critical interfaces are marked red in Figure 4.9 Safety critical interfaces within the subsystems (AWS, VLS and SM-3) are defined and analyzed as the part of "element" SSP, i.e., single system SSP.

The SSIF met the safety interoperability of "Knowledge / safety requirements of remote operations (interfaces)" and "Feedback (Validation & Verification)". Using SSIF strengths, the safety critical subsystems were defined as a part of this analysis. They are:

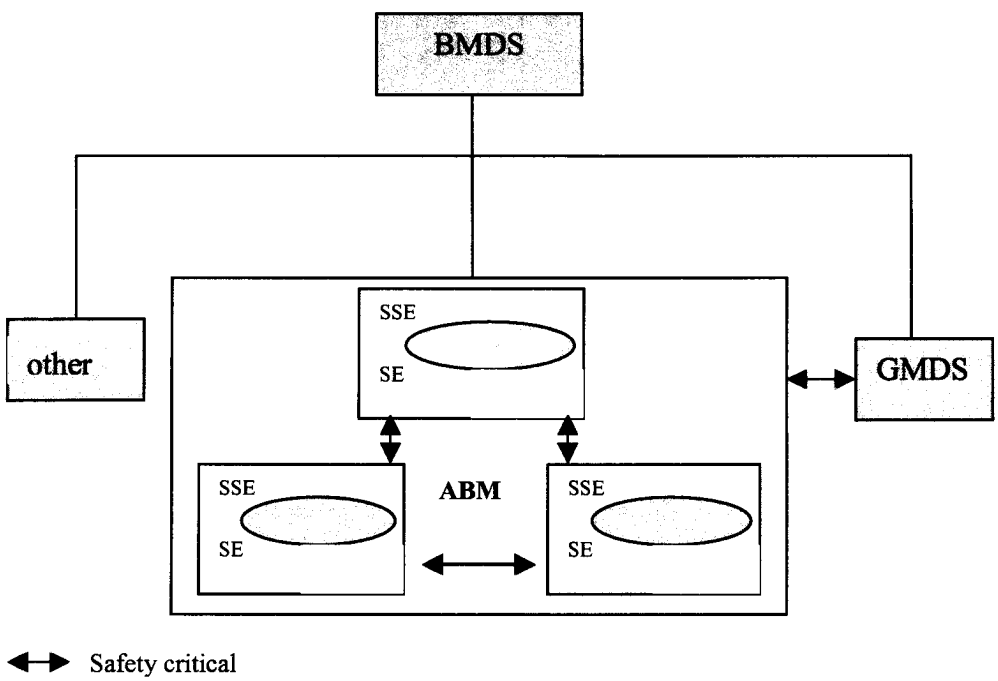

Figure 4.9- ABMD Safety Critical Interfaces

AWS

Safety related subsystems and components:

These are subsystems or components that indirectly are related to the launch sequence functionality. For AWS, Aegis Display System (ADS) is safety related, because it displays the tactical picture, with its status of actions being taken, it assists the operator in making decisions that may have safety implications. ACTS is safety related because if can influence the safety critical functions in respect to Training tracks. Training data shall be separate from tactical data and shall be visibly known to prevent being used as real data. Note that ACTS is not being used in Aegis BMD, and it is merely named here because it is part of AWS.

Safety critical interfaces:

The interface between a safety critical subsystem/component to another is safety critical interface. For ABMD SoS the safety critical interfaces are marked red.

Safety critical subsystems and components:

C&D- it commands and controls engagement orders, it uses IFF function internally or by interfacing with UPX-29 IFF system to ID tracks, it also evaluates threats and designates threats for engagement.

WCS- it prosecutes engagements, it interfaces with the Missile in Flight via SPY radar system, and it can orders Cease Fire and Break Engage.

SPY- it downlinks and uplinks data to the Missile in Flight for guidance.

Safety critical subsystems and components of AWS are shown in Figure 4.10.

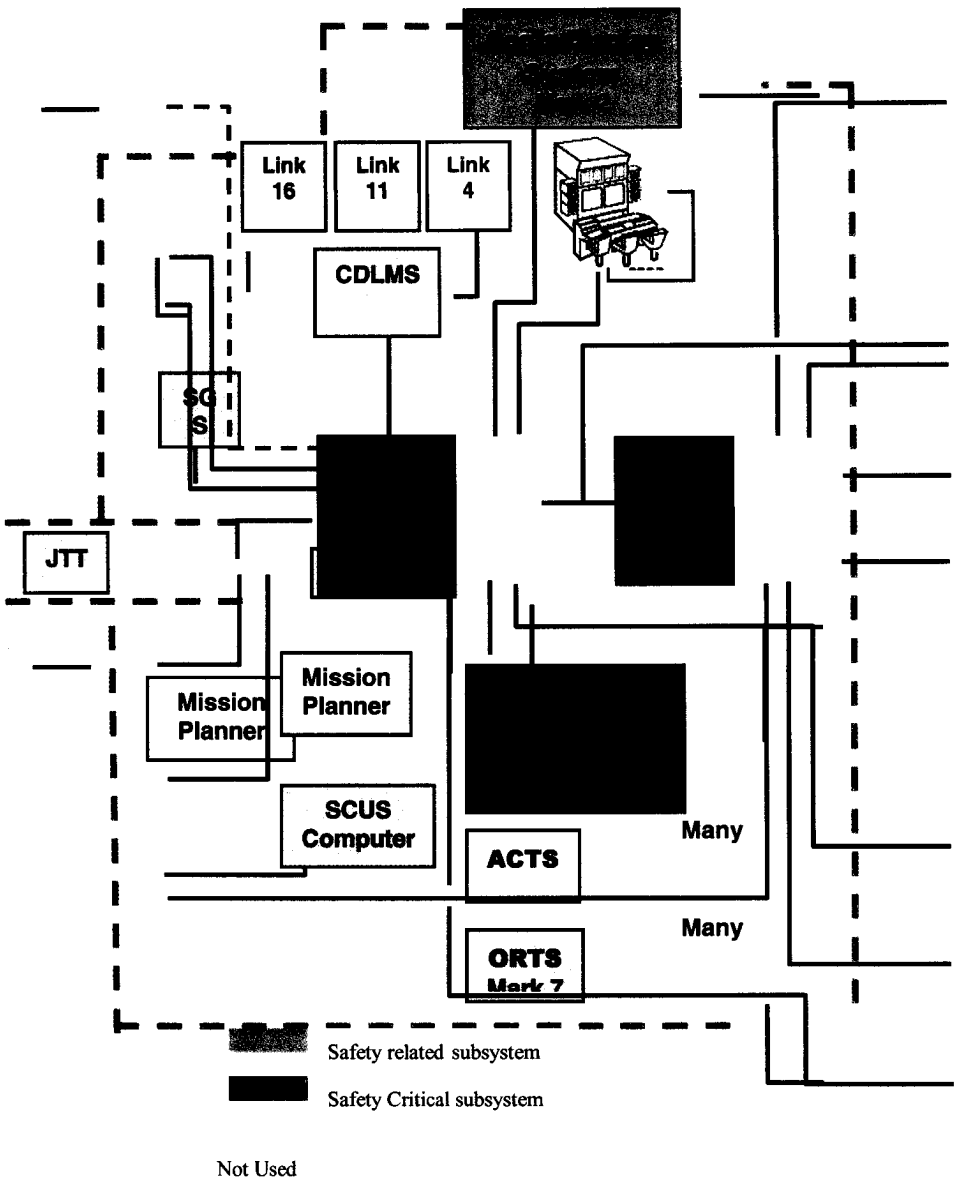

Figure 4.10- AWS Safety Critical Components

VLS

The safety related component for VLS:

VGI- the VGI is part of LCU, and it directly interfaces with the missile, but the data it verifies and validates is out of range time information that results in missile dud. The missile however does not need this data to launch a missile.

The safety critical components for VLS:

LCCP- selects, launches, and safes missiles.

LSEQ- controls the missile interface, commands the opening and closing of the hatch, controls the power supply.

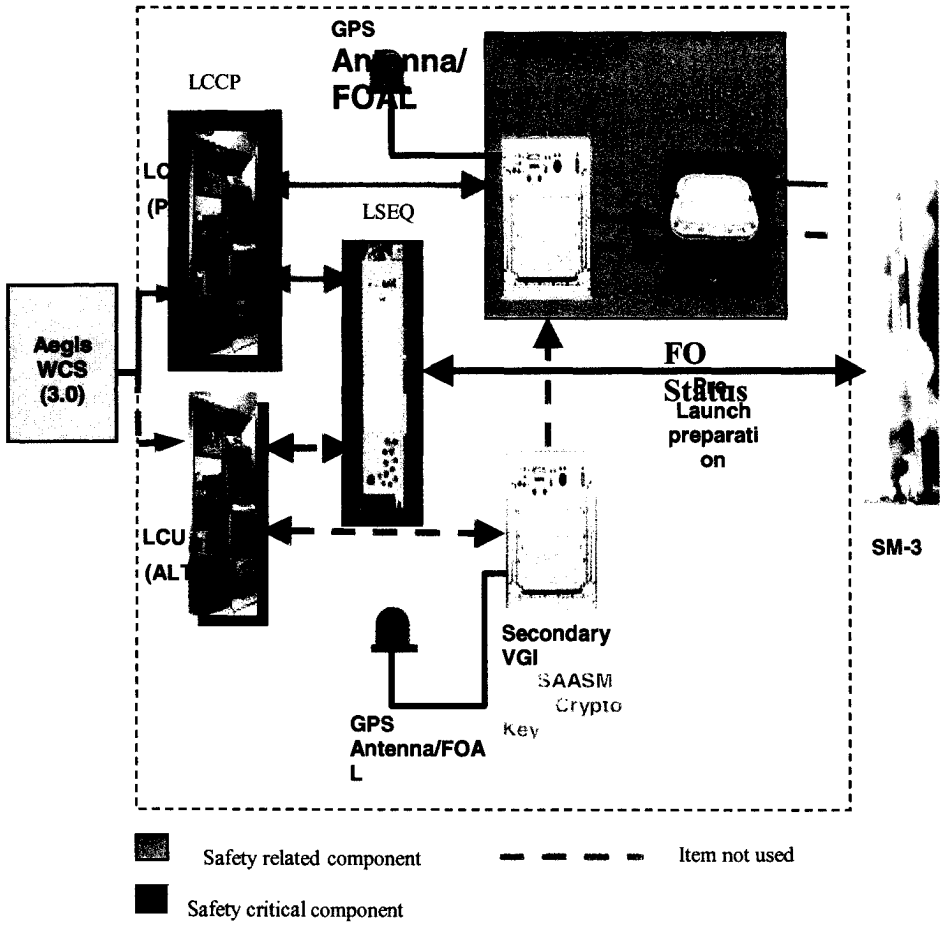

Figure 4.11- VLS Safety Critical Components

SM-3

Safety critical components

As shown in the graph above, most of the SM-3 components are safety critical--KW, DTRM, TSRM, and GP. Figure 4.12 shows all safety critical components of SM-3.

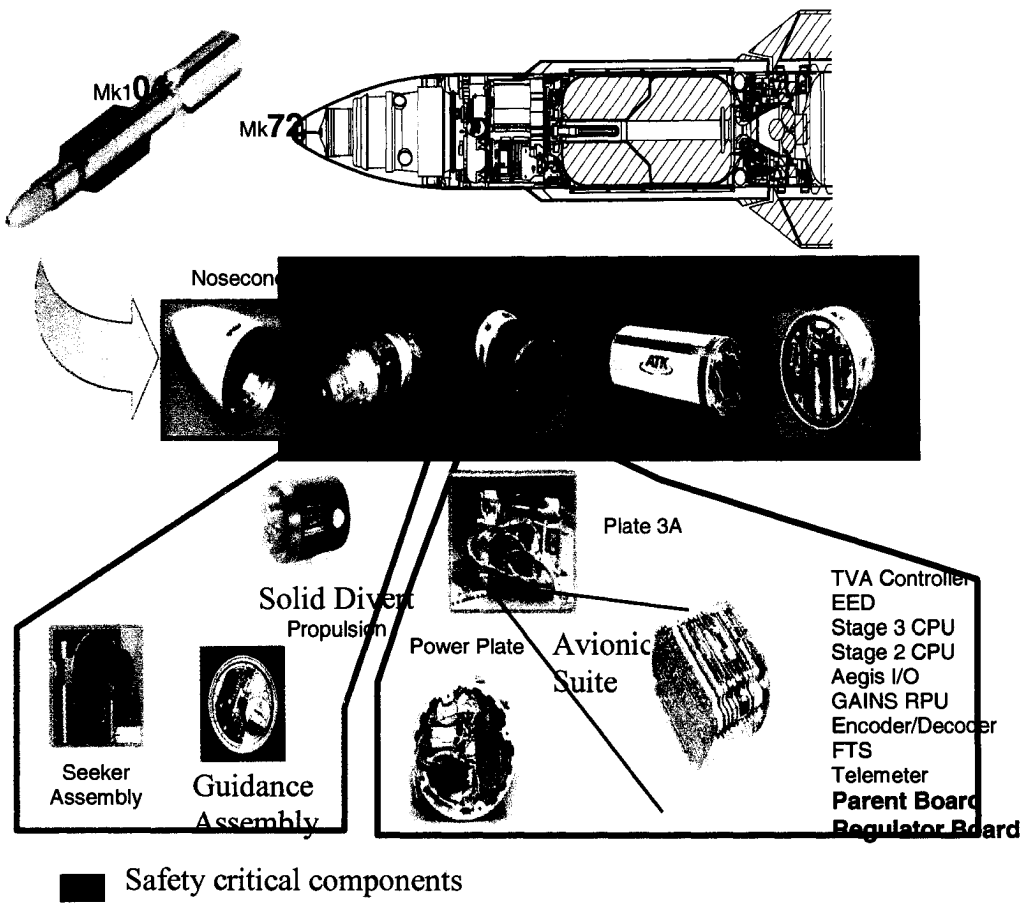

Figure 4.12- SM-3 Safety Critical Components

Aegis BMD SoS Top Level Mishaps (TLM)

TLMs are potential mishaps that may occur during the operation of the combat system. Individual single-system hazards that do not contribute to the overall combat system, is not included in Table 4.4. The potential mishaps are shown and traced to the their associated hazards in subsystems.

Table 4.4- Aegis BMD SoS TLMs

AEGIS BMD POTENTIAL SoS TLMs			
TLMs	**Associated Hazard(s) Originating Subsystem**		
	AWS	**VLS**	**SM-3**
1. Personnel Injury	X	X	X
2. Inadvertent Launch of GMD weapon based on ABMD Data	X		
3. Inadvertent Launch of ABMD Missile	X	X	X
4. Premature/Inadvertent Activation of Ordnance Energetics			X
5. Prosecution of Non-Hostile/ Friendly	X	X	X
6. Damage to Equipment	X	X	X

4.3.4 ABMD SCD

Aegis BMD System of systems SCD are identified below. Note of reminder that SCD portion, as stated earlier, is not a separate tasking; it is embedded as part of SoSSE analyses. The primary reason why it has been highlighted as an entity within the SSIF is that it is the core of safe interoperability interactions. It is the safety critical data that is transmitted, received, processed and performed upon, that makes the safety interoperability an issue to study and to analyze.

Safety Critical Data are the data that cross the boundaries of subsystems (interfaces) and of AMBD overall SoS and have safety implications. SCD 'stems' from safety critical functions (SCF) of the meta-system. SCD can be of different types- messages, variables, positional information, etc.

Error checking data are also SCD for they check for errors in SCD in real time.

As the result of thorough interface analysis, the safety critical interfaces, functions and messages were defined and evaluated for compatibility and verification methods. Table 4.5 shows the analysis results.

Note: Safety critical message are not listed in Table 4.5 in order to keep this dissertation unclassified.

Table 4.5- ABMD Safety Critical Data

ABMD SAFETY CRTICAL DATA (SCD)		
Safety Critical Interface	**Safety Critical Functions**	**Safety Critical Data**
a. AWS (SPY) to SM-3 (only when Missile is in flight) b. SM-3 to AWS	Missile Uplink, engagement Termination SM-3 guidance KW Acquisition and track, aim-point select Missile Downlink	MSL positional data, Break Engage message, System status, command destruct
a. AWS to VLS (WCS to LCU) b. VLS to AWS	Pre-launch, Vertical launch Management, Engagement Orders/Status, Missile Initialization, missile selection, Flight target data	Launch Enable, Fire Inhibit Switch, VLS Magazine Authorization/Permission, Launch Sequence, VLS Mode and Configuration, Missile Inventory, VLS Fault, VOTS, multiple missile launch separation times, Booster Avoidance Processing, and Safe Missile orders, magazine authorization
a. VLS (LSEQ) to SM-3 b. SM-3 to VLS	Pre-launch, missile initialization, Flight Target Data processing	Initialization data, ready to launch message.
a. AWS (C&D, SPY) to GMD b. GMD to AWS	C&D Doctrine Qualification, Engagement Coordination, Threat Evaluation, Identification, Engagement Order, Engagement Monitoring Engagement order	Target data, IFF data, Training /Test data

↑ Analysis of Design

↑ Verification after implementation

SCD Error Checking

A robust program with safety critical application, typically checks itself for errors during run time. Some error checking types look for the length of the message, e.g. 32 bit vs. 16 bit, some will have checksum exceptions set. The legacy program use verification and validation based on the type of message, e.g., the command for launch of a Tomahawk missile may be Message type 20, but the command for launch of a SM-3 may be message type 100, and a wrong message type (based on configuration and the threat posture, etc.) will be rejected safely and without an inadvertent shut down or loss of an interface. The analysis of Aegis BMD revealed that this meta system uses message type verification and validation for checking errors during system operation.

Training / Test

Per Table 4.3.b, one of the SCDs being interoperated by other receiving systems is Training or Test data. Recall Table 4.4. The TLM#5 is " Prosecution of Non-Hostile/ Friendly" and TLM#2, "Inadvertent launch of GMD weapon based on AWS data". When this SCD was applied to ABMD, it was revealed that when ABMD conduct training or test exercise, it does not have a mechanism to distinguish the test or training tracks as simulated; they look like tactical tracks both from symbology perspective and from "bit" perspective. The analysis also revealed that there is no mechanism for the receiving system (GMD) to interpret that data as training or test if it were somehow tagged as training or test data. This was revealed by analyzing the interface design documents between the two systems.

The SSIF application revealed that SCD transmitted to outside the BMD boundary in respect to training or test data cannot be flagged as simulated or be interpreted as simulated by GMD. Therefore, a safety issue exists and needs to be resolved (see the "Findings" section below).

Hazards and Causal factors

As a part of thorough analysis, the analyst (using an SoSSE methodology) must identify the hazards that lead to TLMs. Then the causal factors for each hazard will be identified and risk assessment can be made. These follow-up steps are outside the scope of this research, and will be recommended for future work.

Findings

The current system safety engineering methodology, even when customized and deviated, seems inadequate to analyze the total system. The mapping of SCD in SSIF to ABMD safety critical data (see Table 4.5) reveals that the interoperability issues can be analyzed from system safety perspective using the current method, but applied to all SCD of the system, the methodology seem inadequate. For example, when it is desired to assess the risk in software, the Fault Tree Analysis cannot be useful, mainly because software failures are influenced by number of

interactions, and it is difficult to determine the probability of a failure in the system's life cycle.

It is the recommendation of this research that any methodology developed for system safety of an SoS, be founded on SoS context, safety principles and precepts (listed above), and congruent with development schedule. It is also recommended that a new software analysis tool be created (or the current FTA) be modified to assess the risk of the software.

The inability to send "tagged" simulated data across interface was another significant safety issue. The clear representation of track types for all systems in the meta-system is paramount for the safety of the friendly forces or assets.

It is the recommendation of this research to redesign the AWS so that it would be able to "tag" training or test tracks and clearly represent the simulated tracks to other systems as simulated and totally distinguishable from real tactical tracks.

It is also the recommendation of this research for the receiving systems such as GMD to redesign its system to receive simulated, tagged data for test and training and be able to interpret these tracks as such. The redesign on both sides will prevent any operator confusion and the likelihood for an inadvertent launch of weapons.

4.4 RESULTS

The analysis of the interoperability issues in a specific complex system of systems- namely the U.S. Navy's Aegis Ballistic Missile Defense 3.0 program was performed using SSIF. We applied the core entities of the SSIF, mainly the SoS, SoSE, SoSSE and SCD, to see how we can decompose the SoS, from safety standpoint, to analyze the interoperability data and if we need to, identify the design features that make the SCD's interactions safe. The analysis was scoped to "interface analysis" to evaluate the transmission and interoperation of the tasks by the systems in the combat system. The Safety Critical Data in Aegis BMD was analyzed according to the SSIF attribute. The analysis "framed" the interoperability issues affecting the total combat systems. In at least one case SSIF surpassed the viability of the current process used for Aegis BMD program. When issues discovered, they were documented in "Findings" section following the analysis of a particular combat system function mapped to SSIF attributes.

As the result of the above analysis, and as one of the significant "by-products" of this research, a system safety interoperability statement is produced. This observation represents the "essence" of system safety interoperability, i.e. if we were to plug in the variables (such as transmission, reception, etc.), we would have safe interoperation. The statement is that:
The System Safety Interoperability is a function of: Safe Transmission, Reception, Interpretation, and Performance of Safety Critical Data.

4.5. CHAPTER SUMMARY

The focus of this chapter was the application of SSIF to Aegis BMD 3.0 program for analysis of safety interoperability issues in the ABMD combat system. The Aegis Ballistic Missile Defense 3.0 Program was used as a "use case" to explore the strengths of the framework, and to validate the utility of it for analysis of system safety interoperability issues.

First the ABMD SoS was defined and described. The safety critical subsystems and components and interfaces were identified based on the criteria of paragraph 4.3.3. The top-level potential mishaps, and the subsystems from where the hazards originated were identified. Using this data, we were able to evaluate safety critical functions for the data transmission, receptions, and interpretation. We identified and evaluated all the safety critical data that will be transmitted for actions by other subsystems.

Additionally, the analysis of the interoperability issues concluded with an "emergence" of a system safety interoperability statement. The statement provides insight into the "essence" of safe interoperability concept and allows for more system thinking taking place in each "variable" area. This statement is further response to research question #1 stated in Chapter I.

The application of SSIF to Aegis BMD program and its effectiveness in analyzing the safety interoperability issues is in response of research question #2 stated in Chapter I.

CHAPTER V

CONCLUSION / RECOMMENDATIONS

5.1 OVERVIEW OF THE RESEARCH

In Chapter I, we were introduced to the reality of our times, to our challenging new threats, and our needs to do more to overcome those challenges. We stated that much has been done to use our legacy systems and to synergy our capabilities for a greater good. We also stated, then, proved in literature review, that our performance criteria have left "system safety" behind, and that we should be thinking and saying "safe performance" instead of just "performance". It is a cultural change that we cannot afford to ignore or to delay, for the lives of many can be saved by simply designing safe systems.

This "gap" in our engineering behavior was affirmed by being demonstrated in related literature review, shown in Chapter II.

5.1.1 Review of Research Questions

The purpose of the research was to develop a system safety interoperability framework that can be used to study and analyze the safety interoperability issues in the complex system of systems. In realizing this purpose, we posed two questions that the research had to answer. They were:

- What is system safety interoperability for a complex system of systems?
- How can safety interoperability be analyzed for a complex system of systems?

The first question was answered by:

1. Introducing the concept of system safety interoperability in Chapter I, paragraph 1.6- "safety interoperability", and an example was provided to facilitate understanding of the concept. Then later in the chapter, the system safety interoperability was defined as,

"A capability encompassing many of the safety issues relative to integration, compatibility and interface that are impinging upon the effectiveness with which independent, heterogeneous and/or homogeneous systems, components or elements, including human factor, may safely interact." (Alborzi, 2004).

2. In Chapter II, the safety interoperability was discussed in detail, and another example was provided, in paragraph 2.6.2, "Pre-Mature Launch", involving the interaction between a missile system, a launcher system and weapon control system. As a part of discussion, the system safety issues were provided.

3. In Chapter III, paragraph 3.2 we showed the link between system safety and interoperability and showed clearly that system safety is, in fact, a dimension of interoperability that must be integrated in overall interoperability design and testing for an effective *and* safe performance.

4. In Chapter IV, as the result of the application of SSIF and analysis, a system safety interoperability statement posited that stated that the System Safety Interoperability is a function of safe transmission, reception, interpretation, and performance of Safety Critical Data.

The second answer was answered by:
The Development of System Safety Interoperability Framework (SSIF) (see Figure 3.4.b) by the following procedure:

1. Based on the available literature and lessons learned of some accidents, we developed and established a list of criteria to evaluate the safe interoperability capability of existing analysis approaches in Chapter II. We determined that any method to analyze the safe interoperability capability must meet the following criteria:

A: SoS-based
B: Address of Model Correlation
C: Knowledge /Safety Requirements of Remote Operations (interfaces)
D: Feedback (validation &verification) Capability
E: Translation Methodology (messages, representation of entities, etc.)
F: Safe Federation Extensibility Capability

SoS-based: We stated that the approach used for analysis must be based on systems of systems context. Single-based approaches are unable to address the safety implications, nature and/or the requirements for interoperation between multiple complex systems that need to be logically and functionally integrated and interoperable.

Address of Model Correlation: It was further stated that the approach chosen for analysis must address modeling differences and resolutions/mitigations for differences. Developing a federation of autonomously developed heterogeneous systems involves many real-world entities and potentially many models of those entities by the many systems (Young, 2002). Manual correlation of different models of each real-world entity could be difficult, unsafe, and time-consuming effort. In integrating database systems, the data correlation problem poses one of the biggest safety issues. We mentioned real-world systems that have been having many problems with this issue of correlation anomalies, such as CEC, and it was determined that this criterion will be one of the essential 'ingredient' in resolution of interoperability from safety perspective. Object-Oriented method for

Interoperability is an effective method that can assist the resolution problems, and that SSIF can be founded upon.

Knowledge / Requirements of Remote Operations (interfaces): We stated that an approach to analysis of safe interoperability capability must be able to address interface requirements. First is the knowledge or awareness of the existence or identity of those systems with which it communicates. These requirements include knowledge of interfacing systems' functionalities, a mechanism by which each system in SoS can "talk" and "be understood" by other systems, receive data. This mechanism enables a system within a federation to invoke operations implemented on another systems. Safe-invoking operations is the criterion for analysis approaches. Safe-invoking approach of safety critical command of other systems in the federation is essential for the safe operation of the overall system of systems. Loss of safety critical interface falls in this criterion. For example, loss of uplink data to the Missile in Flight (MIF) is safety critical issue because Missile uses the uplink data to guide itself toward the target. Not having the uplink data as the result of interface loss leads to the MIF flying blindly, therefore causing a catastrophic accident.

Feedback (Validation & Verification) Capability: The next criterion was that any approach must address each system's need and capability to give feedback to other systems in the federation, to validate the safety critical messages as for their format, their content, their size, their acceptable boundaries, etc., and verify that they can be used for operations. It must also address mechanism by which erroneous information is returned to sender without causing corruption or malfunction of the receiving system. Erroneous information include incomplete messages, formatting problem, or "not-understandable' information. The feedback capability is more than a "hand-shake" capability; it verifies that a message can be processed (used) in the receiving system. The analysis approach must allow for managing interfaces by listing and documenting controls, dependencies, and data objects across the interfaces.

Translation Methodology: We noted the two different types of translators being used in engineering community. Early interoperability attempts involved the creation of custom point-to-point (also known as source-to-destination) interfaces between systems. This approach was to resolve representational differences between multiple complex systems, potentially requires n(n-1) translation for a federation of n systems.

The other type was to employ a platform-independent approach in determining the translations during a two-step conversion and that will require 2(n) translations. This method is not only more efficient, but it is more productive in terms of cost and memory consumptions. We mentioned that the safety integrity of safety critical data transmitted between systems increases with less translations, i.e. with 2-step method.

Safe Federation Extensibility Capability: We said that the SoS must be extendable, without reducing its safety quality. This capability refers both to computer languages used in the systems and to extensibility of the federation in terms of additions to the federation. We stated that:

a. A program is considered extensible if enhancements can be made to an existing component or data structure without adversely impacting the dependent entities. Dependent entities are those components that interact with original entity. A federation is safe-extensible when enhancements can be made to an existing component or data structure without compromising (or affecting) the designed-in safeguards in the dependent entities.

b. A federation is considered extensible when additional systems can be added to the federation and changes can be made to information and interfaces among systems without adversely affecting interoperation of the original system federation. A federation is safe-extensible when additional systems can be added to the federations and changes can be made to information and interface among systems without adversely affecting the designed-in safeguards in the interoperations of the original system federation.

This safe federation extensibility provides the foundation for federation reuse (new upgrades, etc.) without concerns regarding safety critical data being impacted.

2. After we established the criteria, we evaluated the two existing approaches (to analyzing the safety interoperability in complex Navy combat systems). We explained and clearly showed why the two approaches cannot meet the criteria and therefore cannot effectively analyze the safety issues in the interoperability of a large complex system.

3. Then in Chapter III, we introduced a new "method" for analyzing the SoS for safe interoperability issues. The new model-based approach was System safety interoperability framework (SSIF). The SSIF was discussed in depth in Chapter III. The core 'elements' or attributes that characterized the SSIF were:

- System of Systems (SoS)
- System of Systems Engineering (SoSE)
- System of Systems Safety Engineering (SoSSE)
- Safety Critical Data (SCD)

SoS- It was stated that when two or more independent, autonomous complex systems are integrated and interconnected as one larger system to perform a higher level of mission requirement, the result is a system of systems. These systems are interconnected and integrated and will be required to perform joint execution of tasks. In fact these systems are now one larger, more complex system, called a system of systems, a meta-system, a Federation (of systems), or

in military terms, a combat system. SoS Engineering will accomplish the integration, interrelation, and interconnection of the systems in the combat system.

SoSE- The SoSE is an evolution (more than an extension) of traditional Systems Engineering (SE). We showed an example of three autonomous systems being required to integrated and interoperate as one larger, more complex system and that the "mechanism" by which this development, modification and reconfiguration or transformation can occur, is SoSE methodology.

SoSSE- We stated that another attribute of SSIF is SoSSE. This is the means by which a meta-system can be analyzed for safety through a SoS Safety Engineering (SoSSE) program. This program will focus on the safety issues involving the meta system. The hazards at the subsystem level have been resolved at the system level within the SSE. The hazards that exist and originate at any of the subsystems and can contribute to the overall risk of the meta system, that is, can contribute to a mishap at the meta system level will be identified, analyzed, and corrected via SoSSE which is an integrated element of SoSE (see Figure 3.4.a).

Moreover, we stated that the SoSSE methodology needs to be developed (left for future research) and listed the number of "characterization" that the methodology must posses for effective implementation of safety program.

SCD- Safety Critical Data is the next core SSIF characterization attribute. SCD is computed during run time by the SCFs interaction through the system of systems, and are analyzed as a part of the SoSSE. SCD exists in all safety critical applications and systems, such as missile systems, command and control systems, explosive systems, and nuclear systems.
Some of the ways to monitor the safety integrity of SCD is:

- Ensure that intended data is correctly accessed (keywords are *intended* and *correctly*.)
- Ensure that SCD is not corrupted
- Ensure the validity of data
- Checking for out of bound parameters
- There is no unused data
- Ensure that there is no memory conflicts
- Ensure error detection/ alerts reporting for SCD
- Safety critical interfaces are identified
- Ensure the integrity of tactical data and training data
- No self-modifying code, especially in safety critical functions
- No unused or dead code
- Ensure that Exceptions are processed correctly

4. Then SSIF was evaluated against the same criteria that we had established in Chapter III, paragraph 3.5, and proved to meet it and therefore, a potential tool for effective analysis of safety interoperability issues in system of systems environment.

5. The SSIF was validated by three subject matter experts to have potential effectiveness for use in analysis of safety interoperability for complex system of systems. Appendix "B" shows the formal memos received from these experts and their short biography.

6. SSIF was then applied to analysis of Aegis Ballistic Missile Defense 3.0 Program. We used the core characterization attributes of SSI in the Aegis BMD program and were able to identify the safety critical functions of the overall meta system and isolate the SCD that were going to be used for interoperations. The application of SSIF to ABMD is shown in Figure 4.3.g

As stated in Chapter IV, paragraph 4.3, and shown in Figure 4.3.b, the subsystems and components of subsystems were identified and analyzed for their safety criticality. Once the safety critical subsystems, interfaces and components were identified, the list of potential Top Level Mishaps (TLM) was developed. As a part of SCD application segment, the safety critical functions were identified and evaluated for the SCD that it will host in order to interoperate with other systems.

We noted the SCD error checking methods, and evaluated ABMD for this verification, and noted that ABMD used message type form of verification and validation during system operation.

System Safety Interoperability Statement

As a "by-product" of the above research analysis, we posited a system safety interoperability statement. The statement will enable scientist and engineers to discuss, apply to real-world issues, and to advance system thinking in systems safety engineering.
We stated that system safety interoperability is a function of:

- Safe Transmission, Reception, Interpretation, and Performance of Safety Critical Data.

5.1.2 Research Contributions / Significance

Contributions

The list of contributions that this research makes to the body of knowledge was noted in Chapter I, paragraph 1.7.2, and listed here (in three categories) for recap.

The analysis of system safety interoperability has made the following contributions to the "Practice" segment of Body of Knowledge:

a. Identification and evaluation of the safety issues in external interfaces and external interactions provides a basis for focused and orchestrated safety analysis from total system perspective.

b. The analysis of the focused area in project management team will translate into allocating the right resources (both from expertise and funding perspective); this means doing the necessary work with optimum efficiency.

c. Identification of the safety attributes of the system that are affected by the integration into a tactical combat system. The safety attributes are those features, design constraints, etc. that define the safety in system context.

d. The "building blocks" created by this research and the analytical methods used will provide more insight into the safety interoperability issues than has the traditional methods.

e. The System Safety Interoperability Framework (SSIF) can be used for mishaps risk reduction purposes.

f. Risk reduction noted in the paragraph "e" above will translate into saving lives of our war-fighters by reducing and eventually eliminating friendly fire accidents.

The contributions to the "Theory" segment of Body of Knowledge include:

b. Defining and documenting, the safety interoperability impediments in a tactical combat system environment. This will aid the future design engineers in designing safety during development of complex systems.

c. Posit a statement of system safety interoperability for complex combat systems.

d. The advancement of "System Thinking" by proving, through scientific analyses, that interoperability capability is not just a performance issue but also a safety issue that needs to be included in all systemic evaluations. The new "improved" systems thinking will enable us to identify processes that have potential in solving complex problems, will also enable us to have control over the "health" of the system.

Significance

The significance of this analysis lies in its somewhat invisible yet crucial role in development of weapons systems and their use in tactical operations. Primarily, the following lists the significance of study:

a. Originality of Concept- Interoperability is not just a Performance issue, but also a safety issue. Although DoD have initiated the effort in understanding and implementing interoperability requirements in today's Navy complex systems, the system safety dimension of it has never been studied or even acknowledged.

b. A new leap into system safety engineering. Up to this point, system safety engineering had a single-system-based approach. No provisions, tools or processes or standards exist for system of systems architecture. No scholarly analysis of system safety interoperability in systems of systems environment has ever been conducted.

c. Risk Reduction/Management- The analysis with this depth and breath will have significant impact on how engineers design safety into the systems, so safety can become the property of the meta-system. Safe systems have inherently low risk for mishaps. This will translate into lower number of friendly fires, inadvertent launch, or launch based on misinformation, miscommunication, etc.

d. System Safety being advanced in "systems thinking". The advanced systems thinking will lead to development of more comprehensive systems theories, application methods and finally in higher systems thinking.

5.2 RECOMMENDATIONS

5.2.1 Development of SoSE / SoSE Methodology

1. System of Systems Engineering discipline to be formally developed for the engineering of system of systems. Currently, in U.S. Navy, the traditional system engineering discipline is used, through 'ad hoc' modifications, as the need arises, to engineer, modify and reconfigure the complex systems.

2. To include Object Oriented Method for Interoperability (OOMI) as part of system of systems engineering methodology. This will enable the development team to resolve the modeling differences in heterogeneous system that are the source of impediments for the interoperation of systems.

3. To include Holistic Framework for Software Engineering (HFSE) as a part of system of systems engineering methodology. The HFSE enables the developers to

keep a requirement-based holistic knowledge of efforts throughout the development phase and beyond. Also the HFSE assists in keeping the "relational" meaning of requirements throughout the development of software. This will especially be beneficial to both the software engineering team and the system safety engineering team.

4. It is recommended that any future studies of interoperability to include " How to measure the effectiveness of interoperability." This is different than "How to achieve interoperability" in that by setting criteria for interoperability and meeting them, it does not mean we achieved interoperability in the level of effectiveness that we require.

5.2.2 Development of SoSSE / SoSSE Methodology

1. It is recommended that System of Systems Safety Engineering to be formally defined and developed for complex safety critical applications. This development must be based on system safety principles and precepts that were mentioned in this research.

2. New analysis tools must be developed to respond to the new system problems in SoS. The current analysis tools that system safety uses (see Chapter II, paragraph 2.3.7) such as FTA or FMECA are inadequate to effectively address the safety issues in system of systems environment. The development of these new tools must be software-based and software-focused to effectively identify the safety issues in interoperations and propose solutions for resolution of conflicts.

3. Although the purpose and the objective of this risk did not include risk assessment, based on the review of current literature (see entire Chapter II), it is recommended that future research efforts be focused on the development of criteria for risk assessment beyond what is provided now. Currently, the risk assessment is based on two categories- probability and severity. It is the recommendation of this research to evaluate the possibility of the third category, such as the "control of software" over the operation of the total system. The re-assessment of current risk assessment method in the software-extensive and critical applications, in the more complex system of systems of today, is a must, and doing anything less will be an absolute disservice to our war-fighters.

4. Development of standards and formal guidance documents are another area to where future researches can be directed. The latest "D" version of MIL-STD-882 is "acquisition-reform friendly" instead of "SoS safety friendly". Standards that establish clear and thoughtful system safety principles and precepts, and are the guiding lights of SoS safety engineers who work in an fluid but adaptive development environment can make a great contribution in eliminating or reducing safety risks.

5. To study the potential applicability of SSIF to non-military complex systems with interoperability requirements.

5.3 CONCLUDING REMARKS

As evidenced in Chapter III and IV, and the summary above, the objectives of this research have been achieved by answering the two research questions listed in paragraph 1.4.1, and then again above, paragraph 5.1.1. The answering of the research questions was made possible by the research purpose—i.e., by the development and application of a model-based framework to analyze the safety interoperability of complex U.S. Navy combat systems. System safety Interoperability Framework (SSIF) was developed and applied to Aegis BMD 3.0 Program for analysis of safety interoperability issues.

REFERENCES

AEGIS Website (2004). *NSWCDD Aegis program, (2004), Available: https://aegis.nswc.navy.mil (Accessed: 2004).*

Aiken, P., Muntz, A., Richards, R. (1994). *DoD Legacy Systems-reverse engineering data requirements,* communications of the ACM 37 (5), pp. 26-41.

Alborzi, S. (2002). *System safety Analyses- An indispensable Element of Reliability Engineering*, International System safety Conference, Journal of System Safety.

Alborzi, S. (2004). *Safe Interoperability in The Systems of Systems (SoS) Engineering and Operational Environment*, Proceedings of International System safety Conference, Naval Weapons Systems Safety Symposium, Providence, RI, Journal of System Safety.

Amanowicz, Marek, Col. (1996). *Military Communications and Information Systems Interoperability*, Military University of Technology, Kaliskiego 2, 01-486 Warsaw, Poland, IEEE.

ANSI/ IEEE Std. 1002-1992 (1992). *IEEE Standard taxonomy for Software Engineering Standards.*

Basili, V., Perricone, B. (1984). *Software errors and complexity: an empirical investigation, Communication*, ACM, Vol. 27, No. 1.

Batard, J. (1991). *Reunion Interoperabilite, Compte rendu, Comite technique de I'ACERLI.*

Batini, C., Lenzerini, M., Navathe, S. (1986). *A Comparative Analysis of Methodologies for Database Schema Integration,* ACM Computing Surveys, Volume 18, No. 4, pp.323-365.

Benkhellat, Y., Siebert M., and Jean-Pierre Thomasse (1993). *Interoperability of Sensors and Distributed Systems,* Sensors and Actuators, 37-38 (1993), pp.247-254.

BMD, 2004. *Integrated System Safety Management Plan (ISSMP),* Ballistic Missile Defense Program.

Center for Army Lessons Learned (CALL), (2004). Available: http://www.call.army.mil/homepage/fratricide.html, 2003.

Chen, B. H., Ling Lang (1995). *Design, Testing and Verification of Safety critical Software, Hazards Prevention,* 4Q95, Vol.31, No.4, pp.22-28.

Chia-Chu, Chiang (2001). *Wrapping Legacy Systems for use in Heterogeneous Computing Environment,* Software Development, ASG Inc., information and Software Technology 43 (2001), pp.497-507.

Chytka, Trina Marsh (2003). *Development of an Aggregation Methodology for Risk Analysis in Aerospace Conceptual Vehicle Design*, a Dissertation, Old Dominion University.

Dean, E. B. (1992). *Quality Function deployment for Large Systems*, proceedings of the 1992 International Engineering Management Conference, Eatontown, NJ.

DoD (2002). *DoD Guidance for System Safety Integration into Systems Engineering Process,* Defense Acquisition Desk book Transitional Site, Available: http://deskbooktransition.dau.mil (Accessed: 2004).

DoD (2003). *Missile Defense Agency Assurance Provisions (MAP), Draft*, U.S. Department of Defense.

Ferry, T. S. (1984). *Safety Program Administration for Engineers and Manager*, Charles C. Thomas Publishing, Springfield, Ill.

Gamble, R. L., Payton D. (2001). *The impact of component architecture on interoperability,* University of Tulsa, OK, The Journal of Systems and Software.

Gibson, J. (1991). *How to do Systems Analysis- An Introduction to Systems Engineering,* Unpublished Manuscript.

Gill, J. (1991). *Software Safety Analysis in Heterogeneous Multiprocessor Control Systems*, Naval Post Graduate School, IEEE, Proceedings Annual Reliability and Maintainability Symposium, pp. 290-294.

Haimes, Yacov Y. (1998). *Risk Modeling, Assessment, and Management*, Wiley Series in Systems Engineering.

Hamlet, D., Voas, J. (1993). *Faults on its Sleeve; Amplifying Software reliability Testing,* Proceedings of the 1993 International Symposium on Software Testing and Analysis, pp. 89-98, ACM Press.

Hammer, J., McLeod D. (1999). *Resolution of Representational Diversity in Multidatabase Systems*, Management of Heterogeneous and Autonomous Database Systems, Morgan Kaufman.

Hammer, W. (1972). *Handbook of Systems and Product Safety*, Prentice-Hall, Inc.

Hansen C. M. (1915). Standardization of Safeguards. *In Proceedings Fourth Safety Congress*, pages 140-146.

Harn, M., Berzins, V., Luqi, and Kemple, W. (1999). *Evolution of C4I Systems*, Proceedings of 1999 Command and Control research and technology Symposium, United States Naval War College, pp. 1361-1380, Newport, RI.

Herrmann, D.S. (1999). *Software Safety and Reliability, techniques, Approaches, and Standards of Key Industrial Sectors*, IEEE Computer Society, Los Alamitos, Ca. 1999.

Holowczak, R., Li, W. (1996). *A Survey on Attribute Correspondence and Heterogeneity Metadata Representation. Available:* http://www.computer.org/conferences/meta96/li/paper.html (Accessed: 2004). http://www.call.army.mil/homepage/fratricide.html, (Accessed: 2004).

IEEE Std. 1228-1994, (1994). *IEEE standard for Software Safety Plans.*

Ikram, N., Dr. S. J. Shepherd (1998). *A New Approach Towards Secure IFF Technique,* University of Bradford, Bradford, United Kingdom, IEEE, pp.1013-1017.

ISO09646, (1987). *Information Processing Systems-Open systems interconnection, OSI conformance testing methodology and framework part 1*, general concepts Norm draft, ISO/TC97/Sc21, ISO DP 9646-1, International Standards Organization, Geneva.

Keating, C. (2002). *Systems of Systems Engineering*, Engineering Management Journal, Old Dominion University.

Keating, C., Sousa Posa, A., Mun N. (2003). *Toward a Methodology for System of Systems Engineering,* Old Dominion University, Norfolk, VA.

Keene S. Jr. (1992). *Assuring Software Safety*, IBM Corporation, Boulder, CO., proceedings from Annual Reliability and maintainability Symposium, pp.274-279.

Khoshafian, S., Abnous R. (1995). *Object Orientation*, John Wiley & Sons, Inc., NY.

Kim, W. (1991). *Classifying Schematic and data Heterogeneity in Multi-Database Systems*, IEEE, Vol. 21, No.12, pp 12-18.

Leveson, N. G. (1999). *Safeware, System Safety And Computers, A Guide to Preventing Accidents and Losses Caused by Technology*, Addison-Wesley Publication Company.

Leveson, N.G., Harvey, P. R. (1983). *Analyzing Software Safety,* IEEE Transactions on Software Engineering, Vol. SE-9, No. 5, pp. 569-579.

Ling, L., Calton, P. (1997). *An Adaptive Object-Oriented Approach to Integration and Access of Heterogeneous information Sources,* Distributed and parallel databases 5, pp. 167-205.

Littlewood, B. (1993). *The Need for Evidence from Disparate Sources to Evaluate Software Safety, Directions in Safety Critical Systems*, pp. 217-232, Springer-Verlog.

NAVSEA INSTRUCTION (1997). NAVSEAINST 8020.6D, *NAVY WEAPON SYSTEM SAFETY PROGRAM,* Department of the Navy.

NAVSEA (2002). USS NIMITZ Combat System Safety Program, *Preliminary Hazard Analysis*, Department of the Navy.

NAVSEA (2004). USS RONALD REAGAN Combat System safety Program, Preliminary Hazard Analysis, Department of the Navy.

Perry, D., Wolf, A. (1992). *Foundation for the study of Software Architecture*, SIGSOFT Software Engineering Notes 17 (4), pp. 40-52.

Pressman, R. (2001). *Software Engineering- A Practitioner's Approach,* McGraw-Hill, Boston, MA.

Pridmore, J., Rumens, D. (2000). *Interoperability- How Do We Know When We Have Achieved it*? Software Science Ltd, UK, Plessey Radar Ltd., UK.

Puett, J. (2003). *Heterogeneous Software Development Tools,* A Dissertation, Naval Postgraduate School.

Qing L., McLeod, D. (1993). *Managing Interdependencies among Objects in Federated Databases*, Interoperable Database Systems (DS-5) (A-25).

Reed, M., Arthur D. Little, Inc. (1992). *A Personal Level Identification Friend or Foe System Employing Free Space Optical Communication*, IEEE, pp.275-278.

Roland, H. E., Moriarty B. (1990). *System Safety Engineering and Management,* Second Edition, John Wiley & Sons, Inc., Los Angeles, Ca.

Sage, A.P. (2000). *Systems Engineering*, Wiley.

Sousa-Posa, A., Keating, C. (2003). *System of Systems Engineering Methodology*, Old Dominion University.
Stephans, R. A. (2004). *System Safety for the 21st Century, The Updated and Revised edition of System Safety 2000,* John Wiley & Sons Publication.

Tomahawk Weapons System (2002). *Afloat Planning System, and Theater Mission Planning Center, Available:* http://www.fas.org/man/dod-101/sys/ship/weaps/tmpc-aps.html (Accessed: 2004).

USN (1986). OPNAVINST 8023.2C. *U.S. Navy Explosives Safety Policies Requirements and Procedures.*

USN (2004). *Missile Defense Agency Management Plan,* U.S. Navy.

USN MIL-STD-882D (2000). *The Standard Practice for System Safety*, U. S. Department of the Navy.

Walsh, A., Couch, J., Steinberg, D. (2000). *Java 2 Bible*, IDG books Worldwide, Inc., Foster City, CA.

Wiederhold, G. (1993). *Intelligent Integration of Information*, ACM-SIGMOD 93, Washington, D.C., pp. 434-437.

Wileden, J, Kaplan, A. (1997). *Software Interoperability: Principles and Practice,* Computer Science Department, University of Massachusetts, Flinders University, Adelaide, SA, Australia, ACM, 0-89791-914-9/97/05, Boston, MA.

Young, Paul E. (2002). *Heterogeneous Software System Interoperability Through Computer-Aided Resolution of Modeling Differences*, a Naval Post Graduate School Dissertation, Monterey, Ca.

ACRONYMS AND ABBREVIATIONS

AAW	Anti-Air Warfare
ABMD	Aegis Ballistic Missile Defense
ACTS	Aegis Combat Training System
AWS	Aegis Weapon System
BIT	Built in Test
BLK	Block
BMD	Ballistic Missile Defense
BMDS	Ballistic Missile Defense System
C&D	Command and Decision
CS PFS	Combat System Principal for Safety
CSSWG	Combat System Safety Working Group
CSU	Control System Upgrade
DTRM	Dual Thrust Rocket Motor
EMI	Electromagnetic Interference
ET&E	Engineering Test and Evaluation
ETBA	Energy Trace and Barrier Analysis
FTA	Fault Tree Analysis
FTS	Flight Termination System
GMD	Ground-based Midcourse Defense
GPS	Global Positioning System
HERO	Hazards of Electromagnetic Radiation to Ordnance
HERP	Hazards of Electromagnetic Radiation to Personnel
IDO	Initial Defense Operations
IPT	Integrated Product Team
IR	Infrared
KW	Kinetic Warhead
LCS	Launch Control System
LCCP	Launcher Computer Control Program
LDO	Limited Defensive Operation
LEAP	Lightweight Exothermic Atmospheric Projectile
LOT	Launch On TADIL-J
LRS&T	Long Range Surveillance and Track

LSEQ	Launcher Sequencer
MDA	Missile Defense Agency
MDS	Missile Downlink System
MEIT	Multi-Element Integration Testing
MP	Mission Planner
MSL	Missile
NSPD	National Security Presidential Directive
NSWCDD	Naval Surface Warfare Center Dahlgren Division
NTW	Navy Theater-Wide
ORTS	Operational Readiness Test System
PFS	Principal for Safety
RADHAZ	Radiation Hazards
RF	Radio Frequency
SCD	Safety Critical Data
SCF	Safety Critical Function
SCI	Safety Critical Interface
SCM	Safety Critical Message
SCUS	System Calibration Using Satellites
SDACS	Solid Divert Attitude Control System
SE	System Engineering
SM	STANDARD Missile
SoS	System of Systems
SoSE	System of Systems Engineering
SoSSE	System of Systems Safety Engineering
SSIF	System Safety Interoperability Framework
SSMP	System Safety Management Plan
SSP	System Safety Program
SSPP	System Safety Program Plan
SSSTRP	Software System Safety Technical Review Panel
TADIL	Tactical Digital Information Links
TBMD	Theater Ballistic Missile Defense
TDDS	Tactical Data Distribution System
TDP	Technical Data Package
TH	Tomahawk
TLM	Top Level Mishap
TSRM	Third-Stage Rocket Motor
VGI	VLS GPS Integrator
VLA	Vertical Launching ASROC

VLS	Vertical Launching System
WCS	Weapon Control System
WSESRB	Weapon System Explosives Safety Review Board

Appendix A

The U.S. Government Release Approval

REQUEST FOR PUBLIC RELEASE OF UNCLASSIFIED TECHNICAL INFORMATION

NSWCDL 5570/1 (REV 01-00) LOG NO:
Ref: NAVSWCINST 5570.1 (series) (PAO Use Only)

1. FROM	2. TO	3. DATE
Alborzi, Showkat S	Public Affairs	

4. DOCUMENT TITLE

A Conceptual Framework for Analysis of System Safety Interoperability in Net-Centric Military Systems

5. AUTHOR	6. CODE	7. TEL NO.
Alborzi, Showkat S	G71	540-653-7828

8. SPONSOR	9. CODE	10. TEL NO.	11. SPONSOR APPROVAL
PD 452 B		(540)653-4464	

12. PURPOSE OF RELEASE

☐ Internet ☐ Presentation ☒ Publication ☐ Abstract

☐ Home Page Address ☐ Contractor Report ☐ News Release ☐ Other

13. CONTRACT NO.	14. TECHNICAL REP.	15. DEADLINE DATE

16. MEETING TITLE (If Presentation, Speech, or Conference Paper)

Dissertation Defense

A. Place	B. Date	C. DoD Sponsored
Old Dominion University	3/29/05	☒ YES ☐ NO

17. NAME OF PUBLICATION (Magazine, Journal, Newspaper) 18. DEADLINE DATE

Doctoral Dissertation

☒ Solicited Manuscript ☐ Unsolicited Manuscript 3/24/05

19. CERTIFICATION: I certify that this document does not contain any classified information.

Author's Signature

REQUIRED APPROVALS: Signature certifies that this material was reviewed for technical accuracy, classified information, security implications, proprietary information, and policy guidelines, and that the subject matter does not fall under the Military Critical Technology List.

20. SIGNATURE (Branch Head)	Release Recommended Remarks:	☒ YES	☐ NO	DATE
21. SIGNATURE (Division Head)	Release Recommended Remarks:	☐ YES	☐ NO	DATE
22. SIGNATURE (Department Head)	Release Recommended Remarks:	☐ YES	☐ NO	DATE
23. SIGNATURE (Security)	Release Recommended Remarks:	☐ YES	☐ NO	DATE
24. SIGNATURE (Public Affairs)	Release Recommended Remarks:	☐ YES	☐ NO	DATE

Alborzi, Showkat S CIV NSWCDL-G71-Branch

From: Drake June C DLVA (DrakeJC@NSWC.NAVY.MIL)
Sent: Monday, March 28, 2005 4:51 PM
To: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Cc: Tim Michael A DLVA; Friedman, Gary H CIV NSWCDL G71-Branch
Subject: FW: Safety Thesis

[redacted]

Alborzi, Showkat S CIV NSWCDL-G71-Branch

From: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Sent: Wednesday, February 23, 2005 11:56 AM
To: Drake June C DLVA
Cc: Cobb, Bradley R CIV NSWCDL-G71-Branch
Subject: RE: PhD Proposal

Tracking: **Recipient** **Read**

Drake June C DLVA

Cobb, Bradley R CIV NSWCDL-G71-Branch Read: 2/23/2005 12:10 PM

June,
In reference to my previous email and voice mail, I am sending you the BMD chapter for Capt Grecco's review. Since I have completed my objectives, I thought he can make a more informed decision based on the final draft rather then just a proposal.
Also, if the Capt would like to review the whole dissertation (in final draft), please let me know. Please note that BMD discussion as a use case is limited to chapter 4 (attached) only.
If the Capt would like a short presentation on the request for approval, I will be glad to do so also.

Thank you for all your help.
Showkat

-----Original Message-----
From: Drake June C DLVA [mailto:DrakeJC@NSWC.NAVY.MIL]
Sent: Tuesday, February 15, 2005 14:05
To: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Cc: Friedman, Gary H CIV NSWCDL-G71-Branch, Cobb, Bradley R CIV NSWCDL-G71-Branch
Subject: PhD Proposal

Showkat

I did talk to Mario who wanted to run ideas past our new CAPT (Grecco) before giving the go-ahead. I've asked him for an answer by end of Feb.

June

PLEASE NOTE: STILL USING LEGACY EMAIL
DO NOT SEND TO NMCI ACCOUNT

June Drake
Work: (540)653-4464
Cell: (540)845-8845
drakejc@nswc.navy.mil

Alborzi, Showkat S CIV NSWCDL-G71-Branch

From: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Sent: Friday, February 18, 2005 10:41 AM
To: Drake June C DLVA
Cc: Cobb, Bradley R CIV NSWCDL-G71-Branch; Friedman, Gary H CIV NSWCDL-G71-Branch; Stoffel, Kevin G CIV NSWCDL-G71-Branch
Subject: RE: PhD Proposal

Tracking:

Recipient	Read
Drake, June C DLVA	
Cobb, Bradley R CIV NSWCDL-G71-Branch	Read: 2/18/2005 10:41 AM
Friedman, Gary H CIV NSWCDL-G71-Branch	Read: 2/18/2005 12:15 PM
Stoffel, Kevin G CIV NSWCDL-G71-Branch	Deleted: 3/15/2005 1:38 PM

June,

Since Capt Grecco has not reviewed my request for approval based on the proposal that I submitted, and since now have the writeup chapter on ABMD use, would it not be better if we can submit that too, and have him review the "actual" ABMD related material, and base his decision on the real product rather than just the proposal that was merely a plan and maybe few pages of AMBD related discussion?

When I sent the proposal and asked for the request of approval to use ABMD as a use case, release it to the Universty and publish the dissertation (for public), the intent was that the final dissertation needed to be looked a also. But since we havent seen a response yet in the past few months, and Capt grecco has not seen or heard my request, and since I already have completed my ABMD related work and the writeup, I thought he would benefit more by seeing the whole thing and base his decision on the "real" use instead of the "plan" to use.

The Chapter on ABMD is 1.66 Meg, (22pages). I cannot attach it to this email, being at home), but if you tell me that this will be a good idea for the time we're in now, then I will go to the office this weekend and email it to you.

Thank you so much for all your effort on my behalf. I will be awaiting your response.
Respectfully,
Showkat

-----Original Message-----
From: Drake, June C DLVA [mailto:DrakeJC@NSWC.NAVY.MIL]
Sent: Tuesday, February 15, 2005 14:05
To: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Cc: Friedman, Gary H CIV NSWCDL-G71-Branch; Cobb, Bradley R CIV NSWCDL-G71-Branch
Subject: PhD Proposal

Showkat,

I did talk to Mario who wanted to run idea past our new CAPT (Grecco) before giving the go ahead. I've asked him for an answer by end of Feb.

June

Alborzi, Showkat S CIV NSWCDL-G71-Branch

From: Drake June C DLVA (Drakejc@NSWC.NAVY.MIL)
Sent: Thursday, November 18, 2004 6:14 PM
To: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Cc: Cobb Bradley R DLVA (E-mail)
Subject: RE: Doctoral Research Proposal

Showcat,

Let me look this over, then I will discuss with the Deputy PD at 452

June

June Drake
Work: (540)653-4464
Cell: (540)845-5845
drakejc@nswc.navy.mil

-----Original Message-----
From: Alborzi, Showkat S CIV NSWCDL-G71-Branch [mailto:showkat.alborzi@navy.mil]
Sent: Thursday, November 18, 2004 1:40 PM
To: Drake June C DLVA (E-mail)
Cc: Cobb Bradley R DLVA (E-mail)
Subject: Doctoral Research Proposal

June
Attached is my Proposal for my PhD research program. It is my desire to use AEGIS BMD BLOCK 04 program as a use case in applying my safe interoperability framework in order to analyze the safety interoperability issues invovled in the system of systems engineering and operational environment. I would like to request the approval of the Program Office for use of the AEGIS BMD artifacts to support my research. The artifacts will be the same that I have access to currently as the AEGIS Safety engineer. No Classified documents or information will be used. No detailed design of any of the elements will be used. The proposal and the upcoming increments of my dissertation will be avaialble for review and scrub by the program office.
The request to use the AEGIS BMD as a use case also includes the publication of my dissertation in May 05.
Please provide me with guidance on how I can proceed with this request.

Respectfully
Showkat
540-653-7826

Appendix B

INITIAL VALIDATION OF SSIF

B.0 INTRODUCTION

The System Safety Interoperability Framework (SSIF) was validated initially (before being applied to Aegis BMD system), by three subject-matter experts (SMEs). Decision-making relative to choosing a design or an approach, by using subject matter experts is not new, however using a structured mechanism for their elicitation did not emerged until WWII (Chytka, 2003). Increased technological innovations and advances in system-based methodologies and thinking along with shortage of experts in the areas of complex systems have renewed the interest in using expert judgment in evaluation of conceptual artifacts.

Expert judgment is merely an opinion based on years of experience, education, research, and lessons-learned. It can be viewed as a snapshot (Chytka, 2003) of the expert's state of mind and knowledge at the time of response to an inquiry.

The definition of expert and expert performance is prerequisite for the assessment that will follow. If there were standards against which to evaluate the expert's judgment, then the accuracy of the expert's judgment will be easy to determine. However, standards rarely exist in the areas of complexity and highly technical problems, which is why only few experts exist in those areas.

Many researchers have tried to correlate experience with expertise. Although, in most cases this may be true, that is, a person with many years of experience in a particular job, would possess an expertise on that area, there is no scientific or statistical evidence to support that correlation.

Some have associated accreditation with expertise. For example if someone has been 'board certified" (Chytka, 2003) or engineers who have passed the "professional Engineer" exam, may be regarded as experts. The issue associated with this thinking is that the certification is a one-time exam, and the performance of the person may decline but he/she could still possess the certification or accreditation.

Peer identification method is that when few experts who are known and considered to be experts are chosen or regarded as experts. Chytka (Chytka, 2003) is concerned that this may be the result of "popularity effect", and that those who may have more insight into the technical problem but are not "popular" or known, may be disregarded as experts.

Chytka (Chytka, 2003) proposes the aggregation of expert opinion in the area of conceptual design (such as SSIF). Her research determines that using three experts can bear useful fruits, and any more will be counter-productive for it leads to less confidence in the final estimate. We will use a modified approach based on Chytka's research.

Approach
Due to the fact that the SSIF is a conceptual framework, thus the confirmation of its viability and effectives infeasible before its application to Aegis BMD program, and the fact that the design of a methodology based on this framework is going to be in the distant future, this "validation" of the experts becomes somewhat subjective. The expert elicitation methodology used in this research is guided by the research of Chytka (Chytka, 2003). The validation of SSIF by the experts uses notions used in Pederson et al. (Chytka, 2003) referenced by Chytka. All of these guidelines have been somewhat modified in this research to meet the applicability of this domain.

For more information of the entire discussion of SMEs and methodologies used, the readers can obtain the cited reference.

Due to specialized nature of this research, a sample of 30 was determined to be sufficient from which the final three subject matter experts can be chosen. This list was chosen from nationally known System Safety Society contact list. This society is located in Unionville, Virginia. Paragraph B.2.1 shows the general criteria used for this sample.

B.1 DEFINITIONS

The following definitions apply within this appendix:

Experience: The years worked or evaluated work. The evaluation assumes that the work is reviewed thoroughly and is evaluated for sound and quality safety program.
Point: value points given to assist in weighting qualifications in order to perform assessment.
BA/BS: Bachelor of Arts/Bachelor of Science includes a 4-year science and engineering degree.
Ms/ME: Masters of Science/ Masters of Engineering includes a master degree in science or engineering. If a person had two masters' degrees, it'll be given double points.
PhD: Doctor of Philosophy in science or engineering.

B.2 PROCESS

The process to identify subject matter experts included:
1. Development of Expertise criteria
2. Identify a small sample (max 30) of candidates
3. Weight each candidate against the general criteria (see Table A-1)
4. Choose the top 5 candidates
5. Weight each candidate against the "specialized" criteria
6. Choose the final three experts

B.2.1 GENERAL CRITERIA

The following, lists the general criteria that the author used in evaluating 30 candidates and selecting the top five candidates.

- Professional Experience
- System safety Experience
- Education

The general criteria were developed by brainstorming in discussion between the author, the advisors, and broader subject of the dissertation.

Table A-1- General Criteria

GENERAL CRITERIA	Weight Points				
	0-5 yrs	6-10 yrs	11-15 yrs	16-20 yrs	21+ yrs
Professional Experience	1	2	3	4	5
System safety Experience	1	2	3	4	5
Education	BA/BS: 1	MS/ME: 2	PhD: 3		

A sample worksheet from is shown in Table A-2.

Table A-2- Evaluation against General Criteria

Candidate	Prof. Eng. Exp <yrs:pts>	System Safety Exp <yrs:pts>	Education <yrs:pts>	SUM <pts>
G. Friedman	30: 5	30: 5	BS: 1	**11**
P. Rogers	15: 3	15: 3	BS: 1	7
M. Zemore	20: 4	18: 4	ME: 2	**10**
D. Bower	21: 5	12: 3	ME: 2	**10**
K. Stottlar	21: 5	20: 5	BS: 1	**11**
A. Lim	7: 2	7: 2	2 ME: 4	8
B. Cobb	18: 4	8: 4	ME: 2	**10**
P.Eagon	10: 2	10: 2	BS: 1	5

B.2.2 SPECIALIZED CRITERIA

The following criteria constitute the specialized areas that the author used to evaluate the top 5 candidates for selection of the top three experts. These areas of expertise were selected based on the subject covered in the dissertation.

- Weapon System Experience

- BMD Experience
- Missile System Experience
- Command &Control (C2) System Experience
- Launching Systems Experience

Table A-3- Specialized Criteria

Specific Criteria (Specialized Expertise)	**Yes: 1 No: 0**	**Yes: 1 No: 0**	**Yes: 1 No: 0**	**Yes: 1 No: 0**	**Yes: 1 No: 0**

B.2.3 EVALUATION

The five candidates with highest score from Table A-2 were evaluated against the "specialized" areas. The following table displays the evaluation worksheet that determined the final three subject experts.

Table A-4- Evaluation against Specialized Criteria

Candidates	**Specialized Expertise**				
	BMD	Weapon System (C2)	Launching Systems	Missile Systems	**SUM**
G. Friedman	1	1	1	1	**4**
D. Bower	1	1	1	1	**4**
M. Zemore	0	1	0	1	2
B. Cobb	0	1	0	0	1
K. Stottlar	1	1	1	0	**3**

Per Table A-4 above, Gary Friedman, Doug Bower and Kevin Stottlar were chosen to review and validate the SSIF for its soundness and effectiveness in evaluating safe interoperability issues in Aegis BMD 3.0 program.

B.2.4 SSIF Validation Criteria

The three subject matter experts developed their own criteria to evaluate SSIF. The combined criteria (without repetition) is as follows:

- Applicability to SoS Operational environment
- Capture of Interoperation capability
- Lessons-Learned (from Gulf war, Iraq War, and near-miss accidents)
- Professional Experience

B.3 VALIDATION RESULTS

The subject-matter experts reviewed and evaluated the SSIF against the above criteria and all approved it for use for Aegis BMD analysis of safety interoperability issues. The complete transcript of approval and the qualification of the experts are noted in the following paragraphs.

B.3.1 Approval

From: Stottlar, Kevin G CIV NSWCDL-G71-Branch
Sent: Friday, January 21, 2005 4:58 PM
To: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Subject: RE: SSIF for your review

Showkat

I have read your System Safety Interoperability Framework and like your approach. I believe you have captured the essence of what challenges are faced in the systems engineering community in developing and assessing a System of Systems particularly the system safety perspective. Well done.

****

From: Friedman, Gary H CIV NSWCDL-G71-Branch
Sent: Tuesday, January 18, 2005 5:51 PM
To: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Subject: RE: SSIF Validation by Subject Experts

Showkat,

I have reviewed your SSIF diagram and found it acceptable for supporting your proposed assessment of system safety interoperability for a system of systems environment.

v/r,
Gary

~~~~~~~~~~~~~~~~~~~~~~~~~~
Gary Friedman
Aegis BMD Combat System Principal For Safety
Naval Surface Warfare Center Dahlgren Division
Code G71
17320 Dahlgren Road
Dahlgren, Virginia 22448-5100
(540) 653-7803 (voice) DSN 249-7803
(540) 845-0036 (cell)

****

From: Bower, Douglas J CIV NSWCDL-G71-Branch
Sent: Friday, January 21, 2005 2:41 PM
To: Alborzi, Showkat S CIV NSWCDL-G71-Branch
Subject: System Safety Interoperability Framework (SSIF)
~~~~~~~~~~~~~~~~~~~~~~~~~~

Showkat,

I have reviewed the System Safety Interoperability Framework (SSIF) and concur with its application to the proposed system.

Douglas J. Bower
Computer Scientist - SSSTRP Chairman
Naval Surface Warfare Center - Dahlgren Division - D72
Office: 540-653-8933 FAX: 540-653-8453 Mobile: 540-845-1929

B.3.2 Biography

A short biography of each of the three subject-matter experts is provided below.

- Kevin Stottlar is currently working for the U.S. Navy as the Aegis Combat System Principal for Safety (PFS). He has oversight responsibility over Lockheed Martin and other support contractors to ensure the safety of the systems used in Aegis Cruisers and Destroyers. He has 22 years of System Safety experience, and has served as Principal for Safety for Vertical Launching System (VLS) and Cooperative Engagement Capability (CEC). He has B.S. in Computer Science.

- Gary Friedman is currently working as AEGIS Ballistic Missile Defense System Combat System safety Principal for Safety (PFS) for the U.S. Navy. He has oversight responsibility over VLS, AEGIS and SM-3 system safety programs. He is also the PFS for SM-2 and SM-6 programs. He has 25 years of system safety experience. He holds a Mechanical Engineering degree.

- Mr. Bower is a Lead Computer Scientist with the United States Navy, working at the Naval Surface Warfare Center Dahlgren Division in Dahlgren, Virginia and has been serving as Chairman of the Software Systems Safety Technical Review Panel (SSSTRP), since March 2001. He has over 19 years of combined experience in system safety engineering and software engineering. Mr. Bower has specialized expertise in software system safety engineering and missile system safety. He has a MS in Computer Science from Virginia Tech. He also has severed for over 21 years as a Field Artillery officer in the Virginia Army National Guard and is currently a Major serving as the Equal Opportunity Officer for the Engineer Brigade 28th ID (M).

B.4 SUMMARY

The initial validation of SSIF was performed using structured process to identify the first pool of candidates from the nationally known system safety society. Then, a general criteria and specialized criteria were developed by brainstorming between the author and advisors and dissertation subject matters. First 30 people were evaluated against the general criteria. Then the highest five scores were evaluated against the specific (specialized) expertise areas, and three experts were identified. SSIF was evaluated against the expert's criteria and found to be a sound framework to analyze safety interoperability issues in Aegis BMD program.

VITA

The author of this dissertation, Showkat Shanaz Alborzi, is currently employed by the Department of the Navy at Naval Surface Warfare Center in Dahlgren, Virginia. She is the lead system safety engineer for the Aegis Combat System and Aegis Ballistic Missile Defense Combat System Safety programs. She is also working on Japanese Cooperative Research program. Showkat is also serving as a panel member on the Software System Safety Technical Review Panel (SSSTRP)-the board that evaluates the software safety program of the Navy's war-fighting systems. Her professional career includes development, testing, and maintenance of Aegis Weapon System software. She regards her fleet support experience that included training of the crew of the Navy's Cruisers and Destroyers as the high point of her career. She also worked on Advanced Tomahawk Weapon Control System (ATWCS) safety program, and led the safety program for the Air Area Defense Capability (AADC).

Showkat holds a B.A. in Computer Science and Applied Statistics from St. Mary's University, 1989, San Antonio, Texas. She holds a Master's of Engineering in Systems Engineering from University of Virginia, 2001, Charlottesville, VA. She holds her PhD in Engineering Management/ Systems Engineering from Old Dominion University, 2005, Norfolk, Virginia.

Showkat has written several technical papers for International System Safety Conferences, and for the Navy System Safety Symposiums.