Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

IMPLEMENTATION OF A FRAMEWORK FOR VULNERABILITY/LETHALITY MODELING AND SIMULATION

Kim J. Allen
Craig Black

Applied Research Associates, Inc.
Emerald Coast Division
962 W. John Sims Parkway
Niceville, FL 32578, U.S.A.

ABSTRACT

The Department of Defense (DoD) has employed Modeling and Simulation (M&S) tools in Vulnerability and Lethality (V/L) assessments of weapon/target systems for many years. A wide variety of simulation tools exist that are used to conduct specific aspects of analyzing weapon systems effectiveness and/or target susceptibility to blast, fragment penetration, hardened target penetration, etc. Previously, a somewhat natural separation of domains existed for these models among surface mobile, ground fixed and airborne target classes. However, it has become evident that many of the methods implemented as part of their respective simulations have applicability across domains. Where applications provide a concrete solution for a particular problem, frameworks are meant to provide a generic solution mechanism for a set of similar or related problems. This simple concept was the key to the implementation of a reusable, extensible architecture known as the Endgame Framework.

1 THE FRAMEWORK CONCEPT

Application frameworks are not a new concept, however there has been very little focus toward this concept within the DoD, particularly within the V/L community. A brief survey of existing V/L applications highlights the need for reuse of behavior methodologies among such applications. The framework concept provides for reuse among many other features. Key components of a framework are described as follows:

- *Framework Core* – The core of the framework, generally consisting of abstract classes, defines the generic structure and behavior of the framework. Typical core processing elements would include the overall process control scheme, message passing, and low-level functionality that would be common across the solution space of similar or related problems. In this case the solution space of interest is V/L assessment of ground fixed, surface mobile, airborne and/or directed energy domains.
- *Framework Library* – This library consists of concrete components that can be used with little or no modification by the applications developed from the framework. Examples would include warhead objects, material property sets, and simple behaviors.
- *Application Extensions* – Additional capabilities, incorporated into the framework that were built to solve application specific problems. An example would be domain specific behavioral methodology, such as fragment penetration and armor spall models.
- *Application* – The application consists of the framework core, the framework library extensions used, and any application specific extensions needed. Applications built from the reusable common framework elements will provide tailored solutions for a specific class of problems, while maintaining flexibility and extensibility.
- *Unused Library Classes* – Typically, an application developed from the framework will not need all of the classes within the framework library. Only those necessary will be incorporated. This allows the applications developed to be lean and focused, not containing unnecessary features and options which may confuse users.

Framework flexibility is defined in terms of how easily applications can be built and configured to support a variety of different needs within the framework's domain. Extensible features of the framework would provide advanced users the capability to add new features and methods quickly and easily. The areas unique to a specific ap-

plication are de-coupled from the generic framework architecture, and are therefore smaller, more focused in scope, easier to maintain, and easier to validate.

2 INTRODUCTION TO ENDGAME FRAMEWORK

Endgame Framework development began as a risk reduction Small Business Innovative Research (SBIR) project by ARA in 1998, for the US Air Force, Air Armament Center, 46 OG/OGM Eglin Air Force Base, Florida. The primary objective was to design and implement the framework concept as related to the V/L endgame domain. One of the key requirements was that the Endgame Framework provides the capability to support and integrate traditionally separate airborne, surface mobile, ground fixed, and directed energy threat analysis within the same scenario. Thus, the architecture had to be generic, extensible, and flexible. ARA's vision for the Endgame Framework architecture is shown in Figure 1. The *framework core* consists of the following major components:

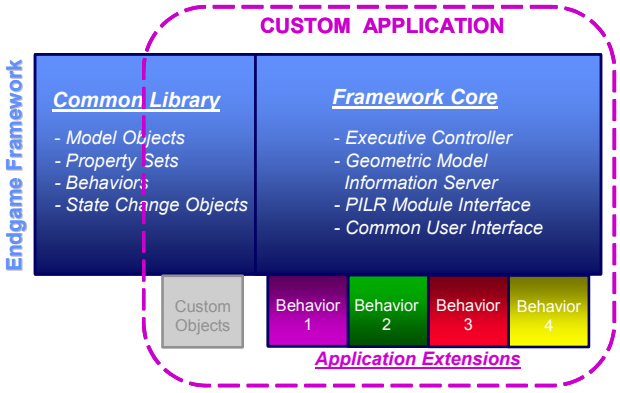

Figure 1: Endgame Framework Concept

- *Executive Controller* – Provides execution control and synchronization through the time-stamped event queue and multi-looping constructs.
- *Geometric Model Information Server (GEOMIS)* – This collection of services provides access to all of the objects in the simulation (target, weapon, scene, etc.) and their properties
- *Dynamic User Interface* – This is a uniquely designed, cross-platform Graphical User Interface (GUI) that frees methodology developers of the burden of updating user interfaces whenever code modifications are made.
- *The PILR Architecture* – PILR stands for Propagation, Interaction, Load, and Response and provides the conceptual infrastructure for application extensions, in the form of behavior methods, which interoperate with the framework and with each other.
- *Common User Interface* – The interface provides users common features and services, and can be tailored extensively to provide application specific capabilities.

Because of its central role in the overall Endgame Framework concept, the PILR architecture and its constituents will be discussed in more detail in the following section.

3 THE PILR PARADIGM

The PILR architecture allows for the break down of complex methodology into simpler components. This decomposition aids the understanding, development, and maintenance of the simulation components. Using the PILR construct generally allows specific, concise behavior methodologies to be assigned to the anticipated interactions for a given encounter. *State Change Messages* are used to allow individual PILR behavior methods to pass the results from one methodology to another. State Change Messages can be viewed as the outcome of the behavior methodologies, and as such are categorized along with the PILR behavior methodologies themselves. For example, the eventual result of propagation, assuming a collision, is an interaction of a damage mechanism with a target component. The result of a simulated interaction between two objects would be a *load* possibly applied to each of the interacting objects. The result of a simulated physical response to the applied load would be a *physical change* applied to the component. Finally, the result of a simulated functional response to the applied physical change would be a *functional change* applied to the component. Individual components of the PILR paradigm are further described in the following subsections. Figure 2 illustrates the PILR paradigm and state change object relationships.

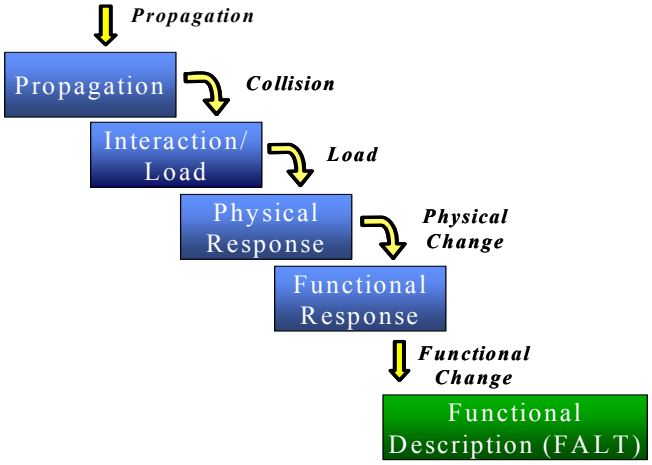

Figure 2: PILR Data Flow

Propagation – This behavior refers to the physical movement of a damage mechanism within or through some medium. Specific propagation methods can be assigned for a particular damage mechanism and model its propagation through a variety of medium, such as a blast impulse propagating through the atmosphere, water, metal, etc.

Interaction – As with the propagation behavior methods, interaction behavior methods can also be viewed as a matrix, or in this case a set of matrices. The interaction methodology can model the effect of a specific damage mechanism interacting with a specific component type, such as a debris fragment and the fuel tank of an aircraft. Alternatively, interaction behavior methods can be defined for a specific damage mechanism interacting with a specific type of material. This more generic case is more typical of current V/L methodology.

Physical Response – The next step in the PILR architecture is to simulate a response to the specific load that was applied. A unique method can be associated with each component type that defines a response for a specific type of load. The result of the physical response method is a State Change Message that describes the *physical change* applied to the component, such as ablation, breach hole, initiation of a rip or tear, or charring.

Functional Response – The last piece of the PILR architecture is the behavior methodology that represents a component's functional response to a specific physical change. The component type and the physical state change message type index the functional response behavior matrix. The functional response method typically involves the application of component fragility functions to determine the probability of component dysfunction or failure for a given type of component with a given type of physical change (damage). This is commonly referred to as Probability of Kill (P_K). Individual components are typically combined using Boolean logic to define the functionality of a complete system or subsystem. This hierarchy is referred to as the Failure Analysis Logic Tree (FALT).

4 THE GEOMETRIC MODEL INFORMATION SERVER (GEOMIS)

The GEOMIS provides a flexible, object-oriented approach to representing, displaying, manipulating, and interrogating geometric models. The geometric models may be imported into Endgame Framework from multiple sources and combined into a single scene for common access. Since the underlying representation of all the geometric shapes are formed from triangles, the GEOMIS can support the generation, display, and manipulation of very complex geometric shapes. These shapes can be generated by using boolean operators such as subtraction, intersection, and union. The GEOMIS also provides wire and pipe classes that can also be used to generate complex surfaces.

The algorithms and methods used to manipulate and interrogate these mesh surfaces are some of the latest methods used in the commercial interactive gaming industry. A combination of the Axis Aligned Bounding Box and Oriented Bounding Box methods are used to efficiently determine ray trace results. As a part of the interrogation methods, the GEOMIS provides the capability to perform collision detection. This feature is a whole new approach to determining component interaction without using ray tracing, which is prone to "miss" collisions between smaller components, such as case fragments and fuel lines or wire bundles.

As an extension of the support for complex shapes and boolean combinations, the geometric models can be dynamically changed to represent changes made to components. This is currently being used to represent component damage such as breach holes, cracks, bending, spall fragments, and debris. The GEOMIS is capable of "on the fly" scaling of a complete model or individual model components. By simply changing the physical properties of an object, such as the length or radius of the warhead, the entire geometric model will be scaled appropriately.

Development of the GEOMIS was performed using standard C++ and OpenGL tools, which has enabled it to easily be ported to several platforms. The GEOMIS currently has been ported to several versions of Windows and Linux operating systems. The GLViewer, which was developed especially for the Endgame Framework, was constructed using Qt, a royalty free, cross-platform graphical user interface tool kit. It provides multiple simultaneous views and a complete set of viewing controls as shown in Figure 3 below.

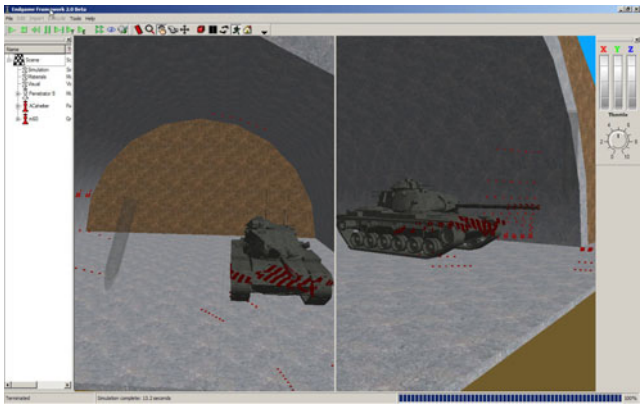

Figure 3: GEOMIS Visualization

5 FEATURES AND BENEFITS OF THE FRAMEWORK APPROACH

Perhaps one of the greatest benefits to the framework approach is that it allows the areas unique to a specific application, termed the application extensions, to be only loosely coupled to the generic framework architecture. This alone allows these behavior methods to be smaller, more focused in scope, easier to maintain, and easier to validate. Reducing the coupling of the methodology to the architecture greatly simplifies the methodology implementation and code maintenance of both. Methodology developers utilizing the framework approach would no longer have to implement and maintain the software infrastructure for "bookkeeping" tasks, execution processing, or much of the user interface, and could focus on development and extension of the behavior methodologies that were important to their applications.

Another important benefit of using a general framework approach is the potential for cost sharing and interoperability among several projects where the framework provides the common architecture. Another benefit is that many V/L applications employ similar methods, if not the exact same methodology. Using the PILR architecture, a common behavior library can be shared among a variety of applications (e.g. basic fragment penetration algorithms or basic laser heating of metals). This would not only reduce duplicative efforts but also promote reuse and cost sharing.

The PILR construct facilitates flexibility and extension of the framework by organizing the needed methodology in a consistent natural way. Incorporating legacy methodology or the development of new methodology typically involves a struggle adapting an existing architecture to the additional requirements of the new methodology. However, the Endgame Framework was carefully designed so that as new methodology is developed or legacy methodology re-engineered, additional properties and state change messages required to support unique methodologies can be created and incorporated into the framework without modification to the framework itself. The use of the PILR architecture, and the state change messages in particular provide yet another, although not so obvious benefit. The State Change Messages applied to the components throughout the simulation are maintained in individual lists as a dynamic property of each component. This information is often exactly what is lacking whenever a synergistic response to multiple damage mechanisms needs to be incorporated in a vulnerability assessment.

Historically, developments aimed at cross-domain support were designed to become monolithic applications with large collections of routines for many applications. As a result they became increasingly large bodies of software, inflexible and difficult to maintain. The Endgame Framework is was designed terms as a technical architecture to avoid software bloat. It contains the basic building blocks necessary to develop a V/L application as focused or as broad as the application integrator chooses. The distribution model of the Endgame Framework allows for developers to deliver complete, compiled applications, including their behavior methodology, or the behavior module itself may be distributed as a compiled linkable library independent of Endgame Framework. With this approach the methodology developers maintain control over their source code and the pedigree of the associated applications. This approach facilitates verification, validation and accreditation by partitioning the methodology into easily accreditable components that leverage a common, verified and validated core.

6 SUMMARY

Modeling and Simulation (M&S) of complex scenarios typically involves the coupling of events that may occur over multiple length and time scales and/or involve stochastic processes that are difficult to model from a purely deterministic approach. To deal with these challenges, the application must be flexible and extensible so that disparate phenomenological models can interoperate and allow for incorporating new or revised models and data. Most legacy M&S applications suffer from one or more of the following difficulties:

- Monolithic; high or redundant maintenance
- Inflexible, limited extensibility
- Expert-friendly
- Data driven, empirically-based, rather than physics-based
- Limited to a single scenario
- No synergy with existing applications
- Geometry-dependent formats
- Platform/OS specific (non-portable)
- Lack of visualization capabilities
- Inconsistent data interfaces
- Lack of Verification and Validation

As programmatic needs and requirements change, the challenge is to extend legacy applications, and/or to combine multiple legacy applications in a seamless fashion.

This often requires significant code rewrite and duplication of effort in order to get disparate applications and their interfaces to interoperate properly. Thus the need to develop more flexible and extensible applications requires some form of template or framework under which all applications may be developed – a type of "meta-application." This approach establishes a standard, robust development environment for applications that rapidly accelerates code development, minimizes coding errors, reduces costs, and allows more flexible interfaces for application interoperability.

Endgame Framework is a very unique application designed to fulfill the "meta-application" goal. It provides several features that have not been available within previous or existing Vulnerability/Lethality (V/L) simulations. Endgame Framework was designed from its inception to be a true framework, which is to say it is an extensible environment with which to develop specific methodology (behaviors), and/or integrate multiple behaviors together to form complete V/L applications, a sample of which are illustrated in Figure 4. Endgame Framework can be thought of as an Integrated Development Environment (IDE), much like Microsoft Visual Studio, specifically tailored for the development of engineering level M&S applications. A few of the features that allow Endgame Framework to stand out among existing architectures are identified as follows:

- Cross-platform support—Windows, Linux
- Cross-domain support—Ground fixed, surface mobile, airborne, directed energy, etc.
- Simulation module implementation as Dynamic Link Library (DLL) or shared object (.so)
  - Literal "plug-n-play" architecture to avoid bloated, monolithic applications
  - Allows complete user control of custom simulation module source code
  - Simple means for distribution of custom simulation modules
- Support for multi-paradigm programming
  - Object-oriented programming allows real-world representation of custom model objects and properties
  - Object-relational paradigm provides the most efficient and extensible implementation of behavior modules
  - Generic programming (templates) used throughout for efficiency

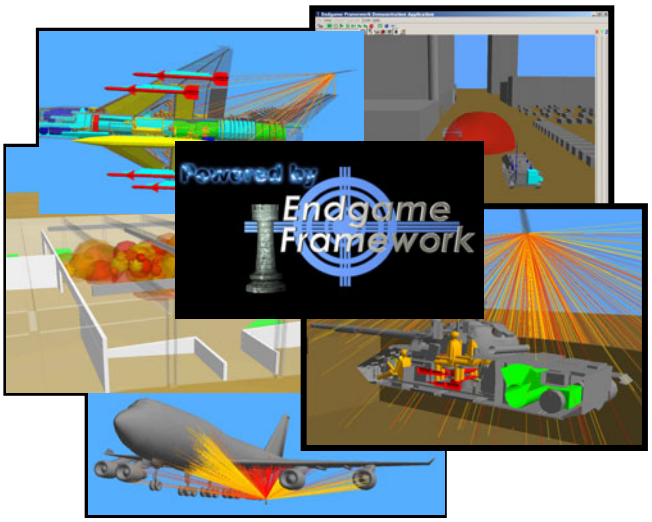

Figure 4: Endgame Framework Sample Scenarios

  - Functional programming supported through Delegates and Signals
- Integrated Geometric Model Information Server (GEOMIS)
  - Supports multiple geometric models, using multiple formats, in a single scene
  - Exceptional visualization
  - Supports traditional ray tracing and optimized collision detection
  - Dynamic deformable geometric representations
  - Complete spatial tree allows dependent and/or independent movement of any object in the scene
  - Integrated geometric model editor
- Integrated Propagation, Interaction, Load, Response (PILR) paradigm
  - Supports higher fidelity physics based solutions
  - Unique/extensible State Change objects key to behavior module interoperability
  - Mechanism to support future synergistic methodology
- Time-stamped event processing
  - Allows interleaving of various events to model multiple simultaneous weapon strikes
  - Support time-stepped methodology
- Fully customizable user interface with supporting properties/property sets
  - Integrated units, bounds, and potential distribution for each numeric property
  - Simple extensible implementation of custom property sets
- Custom model object sharing/reuse

  - Model object factory implementation allows instantiation of custom model objects and property sets across DLL boundaries
  - Automatic serialization (save and restore) of all custom model objects and property sets in an XML format
- Fully integrated interactive execution mode—Start, pause, break, step, etc.

AUTHOR BIOGRAPHIES

KIM J. ALLEN is a senior software engineer at Applied Research Associates. He has lead the development of the Endgame Framework and is currently responsible for its maintenance and enhancements. His e-mail address is kim.allen@ara.com and the Web address is www.endgameframework.com.

CRAIG BLACK is a software engineer at Applied Research Associates. He has been one of the principal architects of the Endgame Framework. His email address is craig.black@ara.com.