Practical Experiences in Creating Components from Legacy Simulations

Judith L. Cerenzia
ARL/PSU
State College, PA
jlc@arl.psu.edu

Thomas B. Haley
NUWC/Newport
Newport, RI
haleytb@Npt.NUWC.Navy.mil

Roy Scrudder
ARL:UT
Austin, TX
scrudder@arlut.utexas.edu

David M. Lounsbury
The Open Group, Inc.
Woburn, MA
d.lounsbury@opengroup.org

Robert P. Goddard
APL/UW
Seattle, WA
Robert_Goddard@apl.washington.edu

ABSTRACT

The concept of creating simulation components which can be rapidly, easily, and effectively composed to meet a range of simulation demands has received considerable interest in the modeling and simulation community. To date, successes in developing simulations from compositions have been limited, especially when these components are extracted from legacy simulations. The Torpedo Enterprise Advance Modeling and Simulation (TEAMS) program developed an approach to create reusable components from a set of commonly used torpedo warfare simulations, and combine components to form simulations to meet specific user needs. This approach combines the features of open architecture development and model driven architectures to establish a common conceptual model and then identify the boundaries and interfaces for the components. Conceptual model development and interface specification use the Unified Modeling Language. Legacy simulation code is then wrapped with middleware to comply with the component specifications, allowing them to be assembled into simulations.

The process described in the paper has been effective in defining and bringing together components developed by multiple organizations in a variety of programming languages and computing environments. The use cases addressed by these compositions range from faster than real-time analysis models, to real-time hardware in the loop stimulation, to detailed engineering analyses that run slower than real-time.

This paper will discuss the conceptual model development approach and how a simulation community of interest can then transition from a conceptual model to a well-defined specification of component behavior and interfaces. We will also describe how this approach leads to both the composition of stand-alone simulations as well as simulations that can be federated with simulations that address other domains. The result is a simulation engineering process that can be applied to any domain to leverage existing simulations to create flexible, reusable components.

ABOUT THE AUTHORS

Ms. Judith L. Cerenzia is a Research Engineer at the Applied Research Laboratory/The Pennsylvania State University (ARL/PSU). Ms. Cerenzia obtained a B.S. in Mathematics and an M.S. in Acoustics from The Pennsylvania State University. She has 18 years of modeling, simulation, and analysis experience in ocean environment, sonar, torpedo DCL, and torpedo defense applications. In support of the above work, Ms. Cerenzia has authored several technical reports and presented technical briefs to Navy sponsors. Currently, she is ARL/PSU's Program Manager for the Office of Naval Research's Torpedo Enterprise Advanced Modeling and Simulation (TEAMS) Initiative. She and the TEAMS consortium are using the Open Group Architecture Framework (TOGAF) Process as a guideline for developing an architecture framework, interface standards and requirements to enable sharing of model components used by the torpedo modeling community. This process is enabling the torpedo modeling community to achieve composable simulation capability using existing legacy components.

Mr. Roy Scrudder is the Program Manager for the Modeling and Simulation Information Management Division of Applied Research Laboratories, The University of Texas. He received his B.A. in Applied Mathematics from the University of Tennessee at Chattanooga Mr. Scrudder is currently working with a variety of modeling and simulation programs and initiatives including the M&S efforts for the Joint Strike Fighter and Multi-mission Maritime Aircraft programs and synchronization of initialization data among the US Army's C4I and simulation systems. He was a member of the Drafting Group and Ballot Resolution Committees for the original IEEE 1516, High Level Architecture for Modeling and Simulation. Mr. Scrudder currently serves as the Product Development Group Chair for the revision of the IEEE HLA standards. He is also an active member of the TEAMS M&S Consortium.

Mr. Thomas B. Haley received his BSEE from the University of New Hampshire. He has 25 years experience at NUWCDIVNPT in design and analysis of torpedo systems, design of hardware, design and implementation of software and software and hardware architectures--particularly real-time systems and DSP, troubleshooting complex systems, reverse engineering, technology transition, modeling and simulation. Formerly as the Torpedo Software Branch Head, he was responsible for all heavyweight and lightweight torpedo guidance and control software design, development and maintenance activities. He also successfully led efforts that passed COMOPTEVFOR OT-IA, OT-IB and OT-IC reviews and audits for the MK 54 (Lightweight Hybrid Torpedo) and for COMOPTEVFOR Software Process Development Reviews (SDPRs) being conducted for the MK 48 ADCAP COT-DV Torpedo development effort, the MK 48 ADCAP Baseline, MODS Block Upgrade Programs. He is currently serving as NUWC/Newport's Principal Investigator for ONR's Torpedo Enterprise Advanced Modeling and Simulation (TEAMS) Initiative.

Mr. David M. Lounsbury received his BSEE from Worcester Polytechnic Institute. He is the Vice President for Advanced Research and Innovation at The Open Group, Inc., where he leads activities related to government research, with a particular focus on developing adaptive and real-time system software. He also served as Vice President of the Collaborative Development Group, which fosters availability and proliferation of open systems technology through collaborative funding and development. Major programs in the group included LDAP, ActiveX Core Technology, DCE 1.2, CDE-Next, and Complex Text Layout PST's, as well as support and consulting activities. Other assignments at OSF include Director of the Distributed Environment Engineering group. This group was responsible for production of the DCE 1.1 and DME 1.1/Network Management Option technologies. David has been the manager of OSF's DCE effort from the announcement of the RFT in 1990. Prior to coming to OSF, David worked for Prime Computer as the manager of the Multiprocessor Operating Systems group, working on systems incorporating CMU Mach and Unix System V release 4 technology. Earlier, he led the Open Systems technology group, which developed a variety of networking products including SNA, TCP/IP, and OSI Ethernet. David is also the holder of three U.S. patents. He is currently serving as the Open Group's facilitator for the TEAMS M&S Consortium, providing expertise for the TOGAF process.

Dr. Robert P. Goddard is a Senior Physicist at the Applied Physics Laboratory of the University of Washington, Seattle (APL-UW). He earned the Ph.D. in Physics from the University of Wisconsin, Madison. Dr. Goddard is the primary author of the Sonar Simulation Toolset (SST) software for computer simulation of sound in the sea. SST is being used at many Navy, university, and industrial laboratories to generate artificial underwater sound, which is used to develop new sonar systems, train sonar operators, and predict performance of new and existing sonar systems and tactics. In earlier projects, he led a group building tools for analysis of biomedical and pharmaceutical data, he served as both a developer and evaluator in a pilot project looking for efficient ways to re-host Navy real-time sonar signal processing applications on commercial off-the-shelf (COTS) hardware and system software, and he served on teams developing a high-performance simulator for passive sonars and a data acquisition system for ocean acoustic surveys. He is an active member of the TEAMS M&S Consortium.

Practical Experiences in Creating Components from Legacy Simulations

Judith L. Cerenzia
ARL/PSU
State College, PA
jlc@arl.psu.edu

Thomas B. Haley
NUWC/Newport
Newport, RI
haleytb@Npt.NUWC.Navy.mil

Roy Scrudder
ARL:UT
Austin, TX
scrudder@arlut.utexas.edu

David M. Lounsbury
The Open Group, Inc.
Woburn, MA
d.lounsbury@opengroup.org

Robert P. Goddard
APL/UW
Seattle, WA
Robert_Goddard@apl.washington.edu

INTRODUCTION

The modeling and simulation community is very interested in the concept of creating simulation components that can be rapidly, easily, and effectively composed to meet a range of simulation demands. To date, there has been little success in developing simulations from compositions, especially when these components are extracted from legacy simulations. The Torpedo Enterprise Advance Modeling and Simulation (TEAMS) program is developing a cross-enterprise, collaborative undersea warfare modeling and simulation (M&S) business environment to enable and encourage the sharing and leveraging of legacy and new components. To achieve the objective of reducing both the technology development time and cost of modeling and simulation tools, and increasing component implementation efficiency and reuse, the TEAMS program developed an approach to create reusable components from a set of commonly used torpedo warfare simulations, and combine components to form simulations to meet specific user needs. This approach uses features of open architecture development and model driven architectures to establish a common conceptual model and then identify the boundaries and interfaces for the components. Conceptual model development and interface specification use the Unified Modeling Language (UML) (Object Management Group, 2005). Legacy simulation code is then wrapped with middleware to comply with the component specifications, allowing them to be assembled into individual simulations that are TEAMS compliant.

The process described in this paper has been effective in laying the groundwork to define and bring together components developed by multiple organizations in a variety of programming languages and computing environments. The conceptual model development approach provides a method for a simulation community of interest to transition from a conceptual model to a well-defined specification of component behavior and interfaces. The TEAMS process also leads to composing stand-alone simulations, as well as simulations that can be federated with simulations that address other domains.

APPROACH

Several organizations have a long history of developing and continuing to evolve modeling and simulation tools for the Navy that are used to develop systems and evaluate system performance in a variety of scenarios. While these organizations developed comparable tools with similar components to satisfy their sponsors' modeling and simulation needs, there was very little communication between them as to which tools were available, and under what conditions they were best applicable. Additionally, several components with similar functionality but varying fidelity were developed in parallel, with no forethought of how to interchange a lower fidelity version with a higher fidelity representation if a sponsor were to desire more realism in a system evaluation. To open the lines of communication among the Navy's modeling and simulation community, two organizations, the Applied Research Laboratory of The Pennsylvania State University (ARL/PSU) and the Naval Undersea Warfare Center, Division Newport (NUWCDIVNPT), were tasked to establish an Undersea Warfare (USW) M&S Consortium. This group of subject matter experts is defining and developing a collaborative USW M&S environment to allow legacy and new components to be interchanged among various system simulations at multiple organizations. Active members of the consortium include the Office of Naval Research (ONR), ARL/PSU, NUWCDIVNPT, the Applied Physics Laboratory of the University of Washington (APL/UW), The Applied Research Laboratories: The University of Texas at Austin (ARL:UT), John Hopkins University/Applied Physics Laboratory (JHU/APL), the Office of Naval Intelligence (ONI), and the Defence Science and Technology Laboratory (DSTL) of the United Kingdom.

Architecture Development Method

Figure 1 shows The Open Group Architecture Framework (TOGAF) Architecture Development Method (ADM) (The Open Group Architecture Forum, 2003), which TEAMS is using as a guideline for mapping out the conceptual model and taxonomy structure, and standardizing interfaces for models used by the Torpedo modeling community.

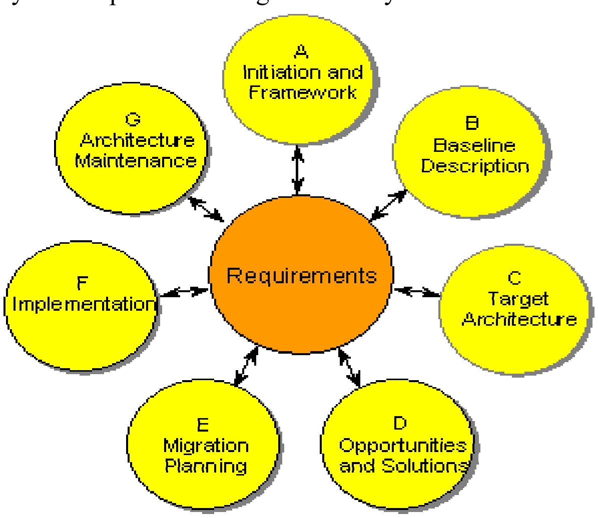

Figure 1. TOGAF ADM

Phase A of the TOGAF process, Initiation and Framework, examines business scenarios to define relevant business requirements, identify concerns of each of the consortium participants, and build consensus among the partners. This first phase was initiated via the Pain Points Exercise that took place during the TEAMS Architecture Workshop May 14-16, 2003, where consortium members each expressed their views on existing issues that might hinder the effectiveness of the TEAMS consortium. The goal of the Pain Points exercise was twofold: First, it served as an interactive discussion among the collected Modeling & Simulation community about problems faced by the members, and served to foster collaboration through an understanding of shared problems. Second, the exercise provides some quantitative guidance on the importance and urgency of the shared problems from the viewpoint of the members.

Phase B focuses on building a baseline description of an existing system, identifying any deficiencies, and compiling an inventory of building blocks currently available. This phase was also addressed during the TEAMS Architecture Workshop May 14-16, 2003, where consortium members each presented briefs on their current modeling and simulation capability. Phase C focuses on capturing multiple views of all needed services to ensure a robust description of a target architecture. In the case of TEAMS, “building blocks” correspond to model components, “target architecture” corresponds to a conceptual model and taxonomy structure, and “multiple views of all needed services” corresponds to addressing consistency of model inputs and outputs that meet the needs of multiple levels of simulation, i.e., systems level through theatre level engagements.

Phase D focuses on Opportunities and Solutions, evaluating and selecting major work packages that will be used to implement the taxonomy structure defined in the conceptual model. OWL Web Ontology Language (World Wide Web Consortium, 2004), Software Interface Design Language (Common Component Architecture Forum, 2004), and other UML and XML (World Wide Web Consortium, 2005) tools are being evaluated to determine the best approach for applying the standard interface requirements of the TEAMS taxonomy structure to the components of existing simulation tools. A combination of the concepts represented in all of these tools is essential to have a graphical, semantic, and textual representation of the TEAMS conceptual model and associated Application Program Interfaces (APIs). The TEAMS Consortium is completing this stage of the process.

Once the tools are chosen and the taxonomy structure and interface requirements are defined, Phase E, Migration Planning, will focus on establishing priorities and developing an outline plan for individual model interfaces that will be developed. Interface definitions for common components across federates (i.e. intra-simulation characteristics) are TEAMS first priority, with a secondary focus on common interfaces among federates (i.e. inter-simulation characteristics) that may interoperate via High Level Architecture (HLA) protocols in an extended scenario (IEEE, 2000). Phase F, Implementation, will focus on an initial proof-of-interface implementation for one common interface specification, as well as executing the full plan for standardizing model interfaces. This will also include verification of the current implementation.

Finally, Phase G focuses Maintenance, establishing a procedure for updating and maintaining the new description of a conceptual model taxonomy structure and relationships, and the standardized interfaces between individual model inputs and outputs. It will also include defining a configuration management plan, documenting the standards, and recording pedigree information for architectural decisions.

TEAMS is in the process of transitioning to TOGAF version 8. TOGAF version 8 expands the scope of the architecture process to include more explicit linkage between the business and mission drivers at the enterprise level, and the components in the technical architecture, as well as guidance on long-term governance of the evolution of the architecture in response to the evolution of the torpedo modeling enterprise.

Conceptual Model Process

The TEAMS consortium began their conceptual model process by defining the initial components of the torpedo modeling ontology for a notional torpedo simulation. Discussions included defining the appropriate components, determining data-flow relationships between components, and establishing how the component structure would fit into the full-scale ontology at a later date.

Next-level taxonomy development expanded the notional torpedo description and fit it into the big picture of a full-scale launch-to-hit simulation representation. Initial subject areas were formed, including how platforms and other entities interact with the environment, sub-dividing platform components and concepts into manageable ontology groupings, and determining how simulation run information and data descriptions would eventually be addressed. Figure 2 shows the initial ontology groupings that comprise the conceptual model of a launch-to-hit engagement simulation.

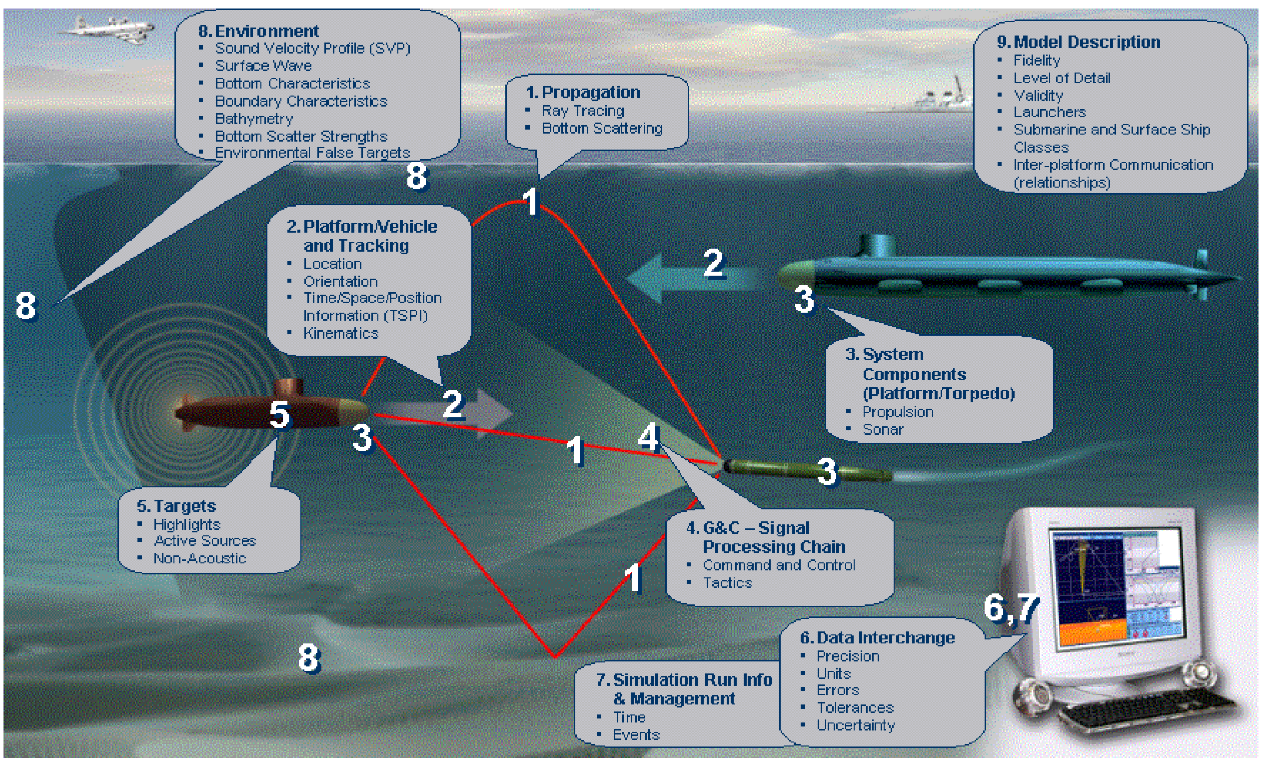

Figure 2. Ontology Groupings

After determining that static representations for the platform, the acoustic environment, and propagation methods could best be captured using Unified Modeling Language (UML) constructs, the consortium began laying out their process for defining a conceptual model. These steps include:

- Define use cases
- Selectively define process and data flow models

- Define major concepts as classes
- Define static relationships between classes
    - Subtypes, aggregations, composition, and general associations
- Define key attributes for the classes
- Define methods that link the classes
- Define dynamic relationships between classes
- Define representative scenarios
- Gap analysis – complete class diagrams with attributes and methods
- Develop additional classes if necessary

Concepts were captured by brainstorming with subject matter experts, drawing diagrams on a white board, then transferring these concepts into UML diagrams.

Use Cases

Use Cases are descriptions that help bound the number of components required for a specific application and the level of detail required for each. The set of use cases the consortium used to determine required components for the conceptual model and ontology framework is shown in Figure 3.

Name	System Emulation of Broadband Weapon	Loss of Array Element	Performance Assessment of Signal Processing Algorithms	Future Weapon System Parametric Analysis	Virtual Torpedo w/ HWIL Pursuing Real Target on Real Range
Objective	Exercise Operational Weapon System That Will Be Tested During Op & Tech Eval	Engineering Evaluation of Robustness of Specific Subsystem Design to Damage	Evaluate Effect of Candidate Algorithms on System Performance Within Fleet - Approved Scenarios Before Next Stage of Design	Assess Performance Metrics, e.g. Pk, TTH Based on Input Environment and Platform Performance Parameters	Cost-Effective Training @ High Realism
Attributes	Not Real-time Highest Detail	High Detail Within Restricted Scope Not Real-time	Fast Results - Many Repetitions & Scenarios (25 Reps/Scenario) Moderate Detail	Simplified Inputs Very Fast Results - Many Reps to Explore Parameters & Distributions (1000's) Low Detail	Real-time HLA Connectivity
Sim Level	Engineering	Engineering	Engagement	Engagement	Engagement

Figure 3. Use Cases

The use cases cover a wide range of fidelity and scope. For example, a detailed analysis to assess how damage to one transducer array element affects performance of the rest of the transducer requires model components defined in enough detail to represent actual elements; treating a transducer as one entity is too coarse. Conversely, parametric analysis to assess feasibility of future concepts only requires simplified components.

IMPLEMENTATION

Conceptual Model

The group of subject matter experts reviewed UML constructs to capture the conceptual model that became the basis for a standardized ontology framework. Static relationships between model components are represented in UML class diagrams, along with some attributes and methods for exchanging data. Discussions for implementing dynamic relationships and control flow between components generated several areas of discrepancy among consortium members, including how to handle the concept of simulation time management, time resolutions, and how to coordinate event-driven components with time-driven components. The consortium decided that time step control was simulation specific, and out of the scope of TEAMS. Control flow focused on event driven components only, and was captured in UML using collaboration diagrams.

UML Class Diagrams
The consortium chose UML class diagrams to represent static relationships between classes within the ontology groupings shown in Figure 2, which include an acoustic environment, characteristics and components of several types of platforms, and models for sound generated by each of the platforms as they interact with this environment. Detailed examples are shown in the figures at the following URL location:

http://uswteams.arl.psu.edu

Clicking on "Reports" will direct one to a set of folders. The "TEAMS I/ITSEC 2005" folder contains class diagrams for an Undersea Warfare Scenario Environment, a Propagation Model, a Generic Platform, and a Notional Torpedo Collaboration Diagram.

Scenarios can take place anywhere in the "world", within a particular medium, such as an electromagnetic or acoustic medium. An acoustic medium can consist of either air or water. The TEAMS consortium captured a description of an Undersea Warfare Scenario Environment in a UML class diagram, focused on characteristics of water as an acoustic medium, and developed class descriptions for models of several components and characteristics of the environment. Characteristics include a water column, a sound velocity profile within the water column, a volume scatterer distribution, boundary information, both simplified and detailed descriptions to support models of variable fidelity, as well as bathymetry, sediment and statistical descriptions of an ocean bottom. Subject matter experts then determined methods with standard sets of inputs and outputs for concepts such as propagating sound, listening, and transforming sound between a source and receiver. Other methods include mechanisms for getting sound speed, bottom type, depth, and additional information about environment characteristics.

The TEAMS consortium then developed UML class diagrams for additional ontology groupings contained within an Undersea Warfare Scenario Environment. First they captured a detailed description of acoustic propagation models in UML. The model's general function is to transform sound. Several types of propagation models are used by the torpedo enterprise, depending on the frequency range and fidelity of an application. Although the consortium recognized that wave models such as normal mode and full-field parabolic equation are used to propagate sound at low frequencies and at a high level of detail, their primary focus for the purpose of engagement modeling was on eigenray models. The UML description includes standard input and output definitions for the propagate method, and a standard set of information used to describe eigenrays, their ray signatures and histories.

The consortium then introduced the concept of a generic platform, and several platform types that could either be a torpedo, submarine, surface ship, aircraft, an unmanned underwater vehicle (UUV), a noise source, or any other vehicle that could participate in an undersea or surface ship engagement.

Classes in this UML diagram represent several of the ontology groupings defined in Figure 2. The class of platform parts represents system components such as propulsion and sensors, warheads, and launchers. Particular emphasis was placed on representing a torpedo, which is an instance of a platform. Additional subclasses are defined for torpedo components to represent tracking algorithms, a tactical controller, and acoustic components of the signal processing chain such as a transducer, beam patterns, beam former, and other sub components.

UML Collaboration Diagrams
UML collaboration diagrams were used to capture dynamic relationships between platforms, components, and an acoustic environment. An example for control flow through a notional torpedo processing system is shown in Figure 4 in the Appendix.

Control flow focuses on event driven components only. Sources located in an environment in the world may transmit or emit energy. The world's listen method passes relevant information to a set of receivers within the world. From a torpedo's point of view, received energy goes through a set of signal processing, detection, and tracking modules. Based on tactical processing, the torpedo makes logical decisions and sends new instructions to its sonar and autopilot to determine which direction it should travel, steer, and possibly transmit energy to further the opportunity of eventually detecting a target. Most methods used to pass standard sets of information between these modules during this dynamic process were previously defined in the UML class diagrams.

Interface Definitions

To ensure consistency among methods that link components across multiple ontology groupings, the consortium next extracted interface definitions from the previous UML diagrams. These requirements will serve as an initial basis from which to begin developing individual standardized interfaces between as many model components as time and funding will allow.

Figure 5 shows initial interface requirements for a notional torpedo, at varying levels of detail based on the previously defined use cases. A top-level torpedo model with standard inputs and outputs can be interfaced directly to multiple federates – the model is delivered as a black box.

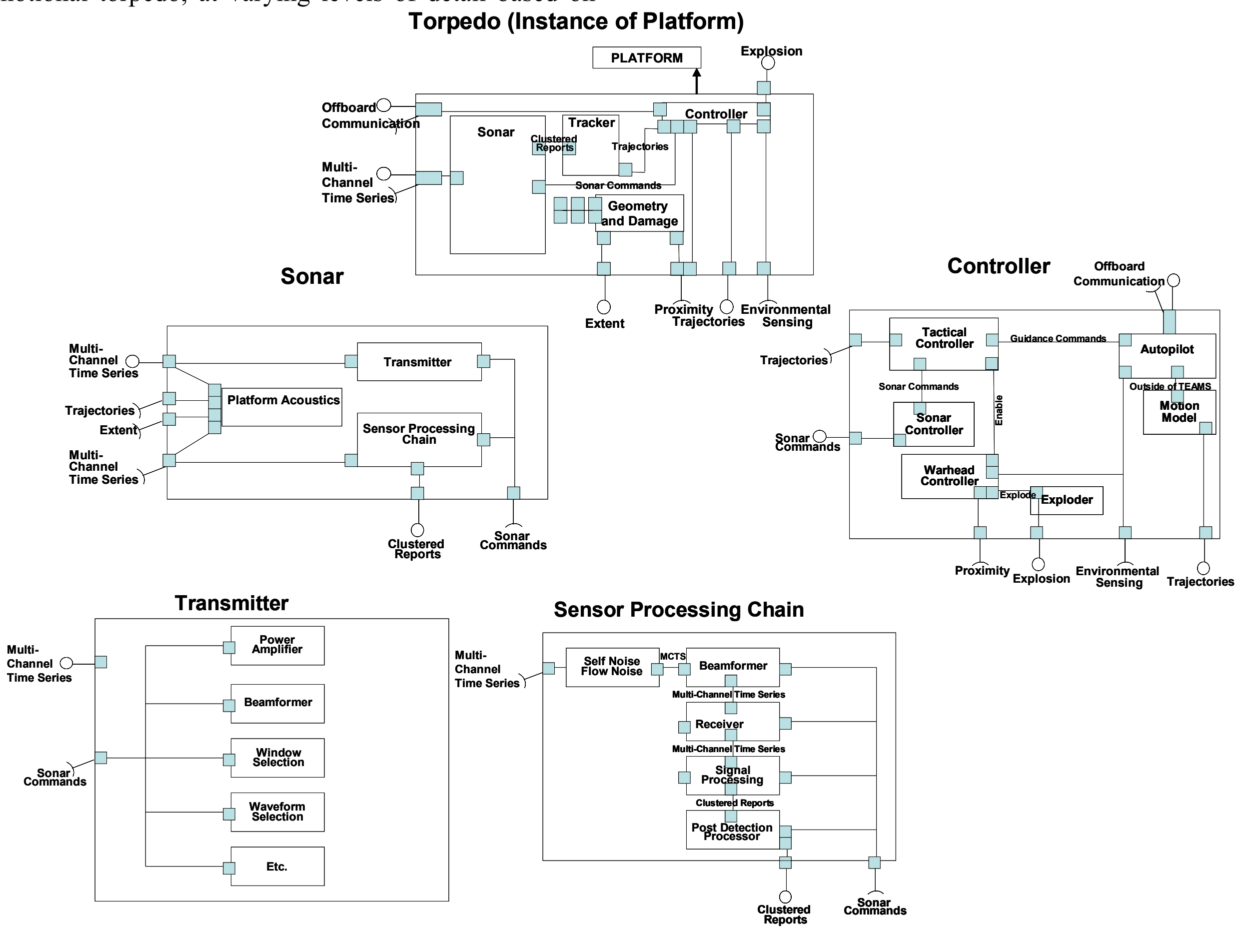

Figure 5. Initial Interface Definitions for Modeled Torpedo Components

At the next level of detail, a user may require a model for a different sonar system, or a modified controller. These subcomponents of the torpedo can easily be interchanged for models with less fidelity, or improved processing capability. The sonar can be further subdivided into a transmitter and a sensor processing chain for received energy. The sensor processing chain contains additional algorithmic components that can be interchanged, such as a beam former or signal processor. Models and sub models that are developed or modified to comply with these standard interface definitions can then easily be used to compose a representative torpedo to evaluate next generation torpedo technology.

FUTURE WORK

TEAMS was initially established to create and execute a cost-effective process to achieve torpedo enterprise intra-simulation interoperability and composability for legacy and future software simulation components. The process created, however, is generally applicable to other domains. In addition to finalizing the interface products identified above for the TEAMS domain, the composability process will be documented in general terms. Although beyond the scope of the original effort, extending the process to facilitate cross-federation simulation composability in support of the High Level Architecture (HLA) Federation Development and Execution Process (FEDEP) would be the next logical endeavor.

Intra-Simulation Composability

The consortium is in the process of performing a proof-of-concept implementation for standard interfaces for propagation and reverberation algorithms. These models were chosen as a first application because they are used widely among the torpedo modeling and simulation community.

An eigenray model used when bathymetry and sediment data are available for specific environments is currently being modified to conform to the TEAMS standard interface for inputs and outputs of a propagation model. A standard interface for a reverberation model that uses non-Gaussian statistics to represent scattering from an ocean bottom has also been incorporated into the TEAMS UML conceptual model, and is currently being implemented into one launch to hit engagement simulation. Both of these models will then be interfaced to additional simulation frameworks to ensure the standards used for the first implementation are robust enough to facilitate true "plug and play" capability.

Standardized interfaces for additional model components, such as target models, intra-simulation components for a notional torpedo, and the rest of the ontology groupings shown in Figure 2 will be developed as funding and time permit.

Composability Approach

A useful proof of concept is described above for a standard interface for an eigenray-based acoustic propagation model. Four simulation systems in current use each contain a different component model for acoustic propagation. Because these four simulations were written independently, there are four different, incompatible interfaces defining the function names, parameter types, units of measurement, and semantics of the function calls for the eigenray model. Each eigenray model also has unique characteristics that could add value if it could be plugged into the *other* simulation systems, which will now be addressed.

One brute force solution to achieve composability is for each simulation developer to force the imported eigenray models to fit into the simulation framework, either by editing the model software (if source code is available) or by writing an adapter layer to translate the function calls issued by the simulation into the form expected by the imported model. If only one simulation developer wants to import the other eigenray models, this approach may be cost effective. However, the scaling law becomes quadratic if all simulation systems want to access features of all eigenray models (perhaps choosing a model best suited to evaluate a particular scenario). The situation worsens because the four simulations (and the four eigenray models) are written in four different languages. The adapter software gets much more complicated and much less portable - each adapter has variants that depend on the particular set of compilers and operating systems for which the components were written. They must also be revised each time the compilers and libraries are updated or a system is ported to another platform. Maintenance gets out of control very rapidly.

The TEAMS approach replaces that quadratic (or worse) scaling law by a linear one. During TEAMS consortium working meetings, the developers of the simulation systems and component models gather in a room and agree on a single interface per component, defined as a "TEAMS-compliant interface", expressed in UML. The UML is translated to language-specific code (a "binding"), such as class definitions and function signatures. Each simulation developer modifies his or her simulation, either by editing it to call the TEAMS interface directly, or by adding an adapter layer that translates the existing function calls to those specified by the binding. Similarly, each model developer either changes his or her component's interface to the TEAMS binding for the component's language, or adds an interface layer to translate TEAMS calls to the original ones. Each software entity (each simulation and each component model) is touched once, either to change its interface to the TEAMS binding or to add an adapter layer.

For simulations and components in the same language on the same platform, the task is complete. The simulation (or its adapter) calls the component (or its adapter) and gets a result in a form it can use. For simulations and components in different languages, another layer is required for two bindings generated from the same UML specification. TEAMS is investigating an approach similar to the Babel (Common Component Architecture Forum, 2003) or CORBA (Object Management Group, 2005) middleware systems, where an interface is defined using an IDL, or Interface Design Language. The middleware includes tools to generate interface code for each supported language, starting from the common IDL. A programmer on the component side implements a component whose calling sequences conform to the "skeleton" interface defined by the binding for the component's language. Another programmer writes client code that calls a "stub" interface defined by a binding generated from the same IDL – but perhaps for a different language. The client calls the stub code in its own language, “magic happens”, the component gets called using its own language’s calling conventions and data representation, it produces a result, more “magic happens”, and the client receives the result. Neither the client nor the component knows anything about the other’s language or execution environment.

CORBA’s “magic” step involves translating data to a common representation (“marshalling”), transmitting it over a network along with a specification of the operation, translating data to a language-specific representation (“demarshalling”), calling a function, and going through the inverse translations to return the result to the client. Babel does the same kind of “magic” without the network, with less overhead for components in the same address space. The cross-language “magic” is done by the middleware or by code generated automatically from the IDL specification, without programmer intervention.

Existing UML modeling tools can generate CORBA IDL directly from the UML. Standardizing on CORBA is an obvious solution, but at the cost in speed due to the marshalling, transmission, and demarshalling steps. Babel’s IDL requirements are very similar to the CORBA IDL, but with much less overhead. Our goal is to find a tool that generates IDL based on the UML models, perhaps via an XML or CORBA IDL intermediate representation. Babel and its Scientific Interface Definition Language (SIDL) appear to provide a promising solution.

A TEAMS-compliant component is one whose interfaces consist of language-specific bindings based on UML models agreed on by the TEAMS consortium. The mechanisms for transforming these components into working, interoperable software look good in principle, but must still be proven in practice. We hope to complete a proof of concept in the next few months.

Federated Simulation Composability

Up to this point, this paper has focused on how UML model development has facilitated defining reusable components that can be composed into simulations. The trend within the DoD and industry has been to move away from large-scale, monolithic, simulation development to either developing single simulations from configurable components (e.g., Objective One Semi-automated Forces) or quite often creating federations of simulations (with the inclusion of interfaces to live systems as appropriate).

The same UML models that led to specifying component interfaces (intra-simulation) can also be used to specify interfaces for simulation federations (inter-simulation). Figure 6 depicts a notional federation constructed from a combination of simulations-- some of which are built primarily from TEAMS-compliant components, others that use TEAM-compliant components but are predominantly non-TEAMS in nature, and other simulations using no TEAMS-compliant components. In this example, High Level Architecture (HLA) has been used to demonstrate combining simulations in a federation. Other simulation integration approaches such as Testing and Training Enabling Architecture (TENA) (Foundation Initiative 2010 Program Office, 2002) would also fit this example, as long as the approach allows the flexible specification of the object model used for simulation communication.

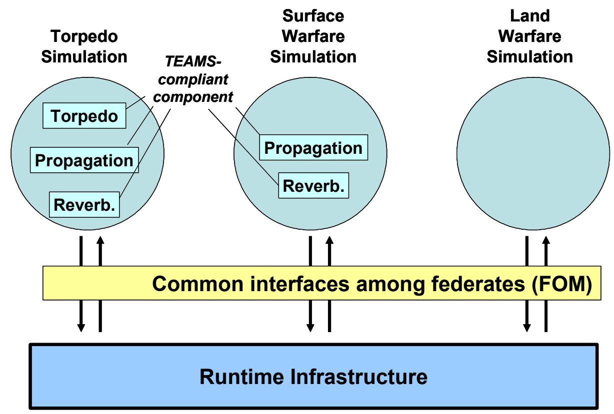

Figure 6. Notional Federation Using TEAMS-complaint Components

In the case of HLA, the Federation Object Model (FOM) specifies the allowable data interchange among federates. FOMs for a federation are typically constructed by merging the Simulation Object Models (SOMs) that represent the data interchange capabilities of individual federates. The UML interface specification shown in Figure 8 could be translated into a set of HLA interaction classes with parameters that could be used in the SOMs for federates that use TEAMS-compliant components. In the future the same interaction classes could then be used in the FOM for a federation that includes these federates.

SUMMARY

Several organizations actively involved in Navy-sponsored modeling and simulation tasks established a consortium to foster communication, collaboration, cost reduction, and process development that allows legacy and new software components to be interchanged among multiple simulation tools at several organizations. The group is using The Open Group Architecture Framework (TOGAF) Architecture Development Method (ADM) to facilitate mapping out taxonomies, ontologies, and interface requirements for components used by the torpedo M&S community. Use cases, UML representations of static and dynamic class relationships, and interface requirements all comprise a conceptual model for a standardized Undersea Weapon engagement simulation framework.

To migrate from a conceptual model framework of interoperable components to actual working code, the consortium mapped out a composability approach for incorporating TEAMS-compliant components into multiple simulation engines. The TEAMS UML provides a platform-independent model (PIM) description. An interface definition language can translate a PIM into a platform-specific model (PSM). Tools such as Babel and its associated Scientific Interface Definition Language (SIDL) and CORBA IDLs without CORBA-specific features are good candidates to provide the appropriate syntax. These tools can also be used to generate program language-specific IDLs for language-specific application program interfaces (APIs), as well as for building standard language bindings. For legacy code, a user will still need to generate interface code (i.e. and Adapter or Façade pattern) for TEAMS APIs, and then generate a language bridge with either CORBA standards or SIDL syntax. The advantage of using SIDL is that translations occur automatically for all language combinations. The above process is executed ONCE per composable component. The model can then be interfaced to any simulation engine associated with the language bridge. A proof of concept of this composability approach will be demonstrated in the coming months. This process can also be extended to allow TEAMS components to federate via DIS and HLA specifications. The consortium will also demonstrate the feasibility of their approach for federations in the future.

ACKNOWLEDGEMENTS

The authors would like to thank Dr. David Drumheller at the Office of Naval Research for continuing to support the Undersea Warfare Modeling and Simulation Consortium, and David Lounsbury from The Open Group, Inc. for helping to establish a consortium that relies on good working relationships among the multiple organizations responsible for USW modeling and simulation. We would also like to thank consortium participants from the Naval Undersea Warfare Center/Division Newport, Applied Research Laboratory/The Pennsylvania State University, Applied Physics Laboratory/University of Washington, Applied Research Laboratories: The University of Texas at Austin, John Hopkins University/Applied Physics Laboratory, the Office of Naval Intelligence, the Defense Modeling and Simulation Office, and the Defence Science and Technology Laboratory of the United Kingdom. The TEAMS initiative would not succeed without their willingness to collaborate to achieve the goals of the program.

REFERENCES

The Open Group Architecture Forum, (2003). *TOGAF (The Open Group Architecture Framework) Version 8.1 "Enterprise Edition"*. Retrieved May 16, 2005 from http://www.opengroup.org/togaf8x/download.tpl?t=pdf&id=106941

World Wide Web Consortium (W3C), (2004), *OWL Web Ontology Language Overview*. Retrieved from June 1, 2005 from http://www.w3.org/TR/owl-features

World Wide Web Consortium (W3C), (2005), *XML Extensible Markup Language*. Retrieved June 1, 2005 from http://www.w3.org/XML

Object Management Group (OMG), (2005), *Unified Modeling Language (UML) Resource Page*. Retrieved June 1, 2005 from http://www.uml.org

Object Management Group (OMG), (2005), *CORBA: Common Object Request Broker Architecture*. Retrieved June 1, 2005 from http://www.corba.org

Common Component Architecture Forum, (2003), *Babel*. Retrieved March 19, 2004 from http://www.llnl.gov/CASC/components/babel.html

Institute for Electronics and Electrical Engineers (2000), *IEEE Std 1516-2000 - IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules.* Retrieved May 31, 2005 from http://standards.ieee.org/reading/ieee/std_public/description/compsim/1516-2000_desc.html

Foundation Initiative 2010 Program Office, (2002), *The Testing and Training Enabling Architecture, Architecture Reference Document, Version 2002.*

Appendix

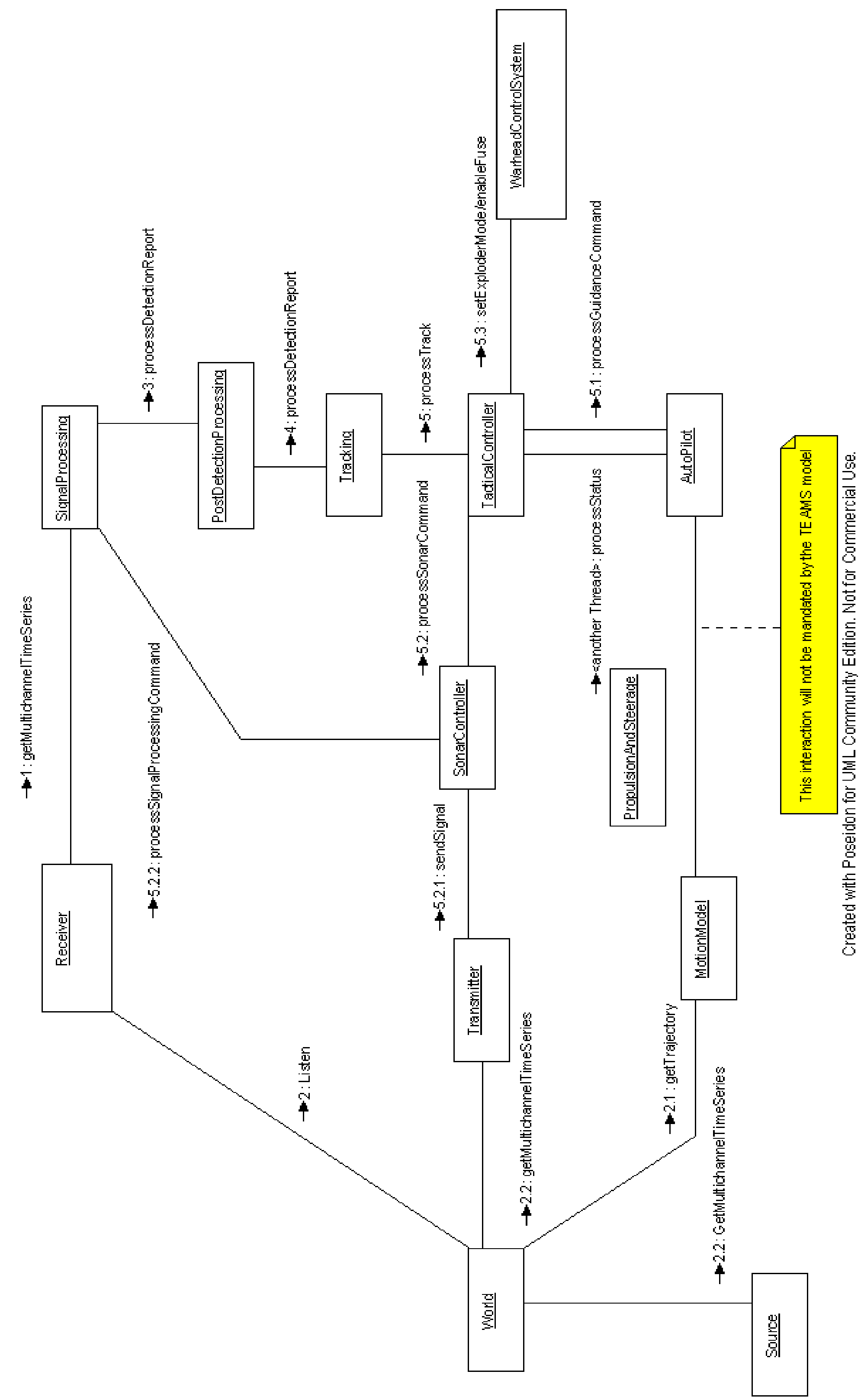

Figure 4. Notional Torpedo Collaboration Diagram.