MODELING AND SIMULATION COMPOSABILITY

Susan Harkrider
U.S. Army STRICOM
Orlando, FL

W.H. (Dell) Lunceford, Jr.
Army Modeling & Simulation Office
Crystal City, VA

ABSTRACT

The term composability is most often used in conjunction with object-oriented software development. In this respect, composability is defined as “the ability to rapidly create or adapt powerful systems to respond quickly to new hardware and software capabilities, dynamically respond to mission requirements, system health/integrity, operating environment, and possibly reconfigure during execution, for example, to trade fault tolerance or security for performance”. [1]

In contrast, the modeling and simulation (M&S) community views composability as an element in the process to achieve automated scenario and exercise generation. In the eyes of the M&S community, composability allows a simulation system user complete flexibility to cross M&S domains or to configure a system “on the fly” from models or pieces of a model.

This paper is written with the express purpose of bounding the concept of composability in modeling and simulation, both what it is and why it is desired. Design requirements are defined, as well as challenges to the process. The paper concludes with a discussion regarding composability limitations within modeling and simulation.

AUTHOR BIOGRAPHIES

Susan Harkrider is a Systems Engineer at the U.S. Army Simulation, Training and Instrumentation Command. Ms. Harkrider is currently the lead systems engineer for the Close Combat Tactical Trainer (CCTT), the Army’s largest distributed simulation system. Her systems engineering experience includes High Level Architecture research and implementation, as well as the Anti-Armor Advanced Technology Demonstration. Ms. Harkrider holds a BSE and a MSIE, both from the University of Central Florida.

W.H. (Dell) Lunceford has been with the Government for 25 years, during which he has been assigned to program management, engineering and R&D projects for some of the Department of Defense's largest simulation based programs. He is currently the Technical Director for the Army Model and Simulation Office (AMSO), which is responsible for shaping and guiding the Army's use of simulation technology. Mr. Lunceford came to this position after serving as the Defense Advanced Research Projects Agency's (DARPA) Program Manager for Advanced Simulation Technology. Prior to that he was both the Technical Director for DARPA's Synthetic Theater of War program which focused on the use of distributed simulation for very large scale theater level training exercises and was the Chief Engineer for the Army's next generation combined arms distributed training system. Mr. Lunceford is a graduate of Florida Technological University and although currently residing in the Northern Virginia area is actually employed by the Army's Simulation, Instrumentation and Training Command (STRICOM) located in Orlando Fl.

MODELING AND SIMULATION COMPOSABILITY

Susan Harkrider
U.S. Army STRICOM
Orlando, FL

W.H. (Dell) Lunceford, Jr.
Army Modeling & Simulation Office
Crystal City, VA

I. INTRODUCTION

The term composability is most often used in conjunction with object-oriented software development. In this respect, composability is defined as "the ability to rapidly create or adapt powerful systems to respond quickly to new hardware and software capabilities, dynamically respond to mission requirements, system health/integrity, operating environment, and possibly reconfigure during execution, for example, to trade fault tolerance or security for performance". [1] Composability, in this context, aims to foster software reuse, typically at the object level or higher. Cited methods for achieving composability include inheritance, delegation, and part-of relationships.

In contrast, the modeling and simulation (M&S) community views composability as an element in the process to achieve automated scenario and exercise generation. In the eyes of the M&S community, composability allows a simulation system user complete flexibility to cross M&S domains or to configure a system "on the fly" from models or pieces of a model. M&S domains include requirements definition, training, and materiel development. Building a generic simulation system from the ground up is extremely expensive and time consuming, and no one can ever anticipate all of the requirements. Because of this, software reuse is necessary, but it does not always fulfill its intended purpose.

When considering the breadth of capabilities expected of simulation systems, it is readily apparent that a single system expected to meet this diverse set of needs should have the ability to be composed. This will support the particular objectives of each training exercise in which it is to be used, while eliminating extraneous processing requirements for unused functionality. The process of composition should address the content of the simulation as well as the resources required, i.e., hardware and personnel needed to conduct the exercise.

To clarify, composability differs from configurability, e.g., generating an exercise that includes eight rather than four M1A2 tanks. In this case the M1A2 tanks are identical, yet have different missions and tactics during an exercise.

II. DEFINITION OF COMPOSABILITY

Managers desire vast functionality within the most inexpensive system possible, and this economy forces the system builder to host a huge model library within the simulation. Composability allows the most efficient use of simulation capability, while maintaining management and user objectives.

Therefore, a derived definition of modeling and simulation composability is: The ability to create, configure, initialize, test, and validate an exercise by logically assembling a unique simulation execution from a pool of reusable system components in order to meet a specific set of objectives.

III. DESIGN REQUIREMENTS

Methods of composability

Given the above, composability can be defined further, based on its usage to satisfy a time requirement versus a system requirement. The requirement effects the method for achieving composability: assembling raw data, extending system components, or exploiting an existing architecture.

Within a given timeframe, building a simulation is achieved by composing a unique system

from a library of existing objects or from raw data. Examples of existing objects are things that are coded and reside in libraries, such as a tank model of a body, tread, engine, and turret, or some similar set of models. Extending system components is the ability to create new federates from a library of models, and is one of many elements in the automated scenario generation process. While the scenario generation process typically requires a full year for a large exercise, the Joint Simulation System (JSIMS) requirement is system composition in 96 hours. Assembling raw data such as mathematical models, algorithms, or knowledge-engineered sources is a second means of composability. At this level, issues concerning validation of the doctrine, tactics, and Simulation Object Model (SOM) arise. Issues associated with these methods of composability include the amount of data required within each model, the terrain implications, and the impact on behavior when a model or model data is reused.

Alternatively, a composable architecture is one in which the simulation system is constructed of an infrastructure of independent layers of software with well-defined application programmer interfaces (APIs). A composable architecture is a near-term requirement, but may not be feasible outside of a particular system specification. In effect, modules from one composable architecture may not be physically nor logically interoperable with another architecture unless the calling interfaces are standardized. An example of this is the architecture in a personal computer. Both the hardware and software interfaces in an IBM PC are standardized to the point that almost any component, whether it is a central processing unit or a word processing program, is easily plugged in and used. This is a very useful feature in a simulation, however the community will be very reluctant to standardize on any one hardware or software option.

Levels of Composability

At what level of implementation is composability appropriate? An example of this is a chemistry laboratory. If a chemist were required to produce any possible compound, what elements would be necessary? Would it be faster to stock the compounds instead? And, if all the compounds could be stored, how expensive would the lab be? Analysis would determine the point at which it would be more effective to stock basic elements versus cheaper and faster to stock the compounds. Similarly, in a planned federation, it would be prudent to determine at which level composability should be implemented in order to achieve the greatest benefit at the lowest cost. In this respect, composability should be separated into four levels: configuration, component, model, or system level.

Configuration - Configuration is the ability to alter the number of identical entities per exercise.

Component Level - Objects internal to a system from which new objects can be built, e.g. ModSAF, is component level composability. The delimiting factor is the requirement for objects, algorithms, and data to be housed internal to a system. (See Figure 1). [2]

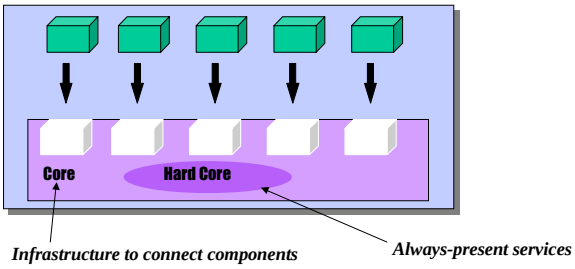

Figure 1. Component Level Composability

Model Level - At the model level, composability is achieved by selecting preassembled packages containing numerous models that represent a consistent subset of the battlespace. Model level composability can also be achieved by assembling suites of hardware connected by a LAN/WAN combination. The interfaces of packages are selected to achieve the greatest possible interoperability with other packages by minimizing the degree of external interaction. (See Figure 2). [2]

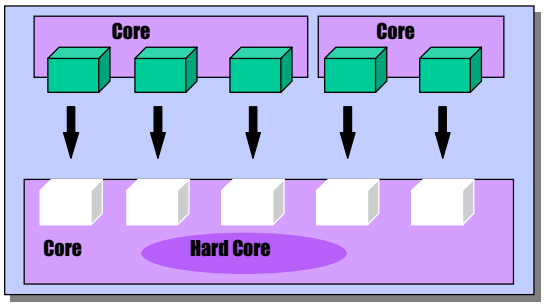

Figure 2. Model Level Composability

System Level - System level composability is achieved by pre-assembling a complete simulation. At this level, the simulation can be modified by parameterization, or by substitution of alternative packages drawn from a limited pool. (See Figure 3). [2]

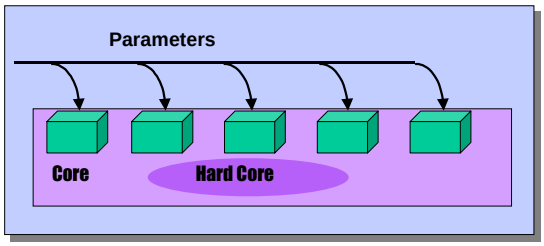

Figure 3. System Level Composability

IV. CHALLENGES

Verification & Validation

Composability requires advances in procedures for verification and validation (V&V). In general, an arbitrary assembly of validated components will not produce a validated simulation. V&V in a composable environment should be as sensitive to simulation context as it is to simulation components. It is expected that there will be stand-alone V&V requirements unique to a particular composition, therefore, the owner of that composition is responsible to accomplish any additional V&V needed to complete the testing program. The presence of metadata concerning models and system components will assist the V&V process significantly. The V&V process requires some form of configuration control that will narrow the scope of the composition and allow for comprehensive testing prior to the exercise. Validation can be reasonably executed at the model level, allowing validated simulations to be composed with the required flexibility.

User interface

Compositions will only work if planned and designed from the beginning to meet specific objectives. Most agree that composability requires some form of a repository to catalog functionality and limitations, and to house the model meta-data. The repository should contain tools to manage combinations of different objects. The sophistication of the tools will depend on the level of composability desired, e.g., a simulation composed at the model level will require a more intelligent tool to assist in assembling an exercise than a tool used to compose at the system level.

Ripple effect

Further, composability can only occur in a well-tested, controlled environment. The system designer should not be able to randomly compose systems that have not been constructed previously nor tested. In effect, the system should not be developed without tools to support testing. The composable environment should offer an intelligent consistency check tool to warn designers of the implications of their creation. For example, if a sensor system is taken from an airframe and moved to a ground vehicle, the tactics and doctrine of that vehicle will probably be affected, as well as the capabilities of the sensor. If the transfer of the sensor is illogical, then the tool should recognize the problem and alert the designer.

Legacy systems

Another concept to be studied is that of including virtual simulations in a composable environment. The concept of composability is currently limited to computer generated forces simulation. Adding a virtual simulation to an exercise assembled from composable elements is not an issue until one considers the potential of dynamic composition. Although none of the program representatives interviewed relayed the need for dynamic composition, their program documentation hints at the requirement.

High Level Architecture

A special case of composable architectures is the High Level Architecture (HLA). Within this architecture, federations exploit a detailed interface specification for passing data. The internal system characteristics and design decisions are not limited by the HLA. Systems that meet the compliance requirements of the HLA have documented their characteristics in a simulation object model (SOM). The SOM class table identifies the base entities available to a system designer, yet does not necessarily represent the object model internal to a federate. Therefore, the HLA is a composable architecture, however it does not offer a ready means to validate models or to validate interactions.

Configuration management

Implications to configuration management are acerbated by composability. As changes are made to achieve a composition, the system baseline requires global updates. This is especially evident in factory level composability. The achievement of composability is not a one-time effort, but will require continuous attention and coordination in the M&S community. As development experience is gained, and new requirements and capabilities of technology to support composability are identified, the definition of composability will evolve. Descriptions should be developed by the community to provide identification of composable products. The descriptions will provide an operational use for the composable product, its functionality, software product properties, relationship to architecture, interfaces, required computational resources, and allocated requirements.

Model Metadata

Various repositories of model metadata have already been developed and will continue to be developed by the M&S community. Metadata describes the aspects of each simulation component such as its associated simulation control architecture, e.g., event or time-based, user requirements, constraints, and algorithms. The metadata is used by the composition environment to assist the user in browsing components and developing a valid simulation or simulation entity.

For example, the discrete event simulation community uses any of several software packages to accomplish model level composition. Arena, for one, offers modules containing specific data and functionality that can be placed within an active screen to perform a discrete event simulation. The modules can be chosen from one of many menus, and can quickly function as a simulation with minimal programming expertise. The modules within Arena are limited in their application, however the capability to compose a simulation rapidly from existing packages of functionality is achieved.

Ideal FOM

The concept of a Federation Object Model (FOM), an “Ideal FOM”, that covers the full spectrum of possible scenarios, could be developed and demonstrated. The Ideal FOM would be limited to a small system, since changes to a FOM require exponential updates to the system. If an Ideal FOM appears somewhat impractical, then an abstraction of sub-FOMs is an alternative. In this case, a casual set of interrelationships between objects could be defined, however the interaction issues would need to be resolved by an intelligent tool. The subset would contain packages of model element information that can be reused in exercises.

V. SUMMARY

As of today, the scope of composability is limited to computer generated forces (CGF). Several programs have defined and delineated composability for their intended use, but have not built a robust composable simulation system. These programs include JSIMS, WARSIM, OneSAF, JWARS, and JointSim. Further, amongst these programs a standard characterization of design components and terminology for M&S composability has yet to be determined. This paper is the first step in the standardization of composability terminology and component identification.

VI. REFERENCES

[1] Composable Systems and Configurable Computing Conference Proceedings, available at http://www.ito.darpa.mil/PIM/wed34.html.

[2] Butler, Brett. *Simulation Composability for JSIMS.* 7th CGF & BR Conference, 12-14 May 1998.

[3] *The JSIMS Architecture*, Version 1.0, 30 June 1997.

[4] *Composability in the JSIMS Domain*: *WARSIM 2000*, unpublished document.

[5] *OneSAF Analysis Phase II Contract Proposal*, Science Applications International Corporation (SAIC), 1 December 1997.

[6] *JointSim Research – Composition of Simulation Objects*, Los Alamos National Laboratory, 23 January 1997.

[7] Carnegie Mellon University EDCS Technical Status Report, available at http://www.cs.cmu.edu/~Compose/html/EDCS/FY98.

[8] Workshop on Composability Issues in Object-Orientation. Part of the Tenth European Conference on Object-Oriented Programming Proceedings, 8-12 July 1996, available at http://wwwtrese.cs.utwente.nl.