

# Umbra, an Object-Oriented Simulation and Integration Framework

Robert Brittain, Joseph Barfoot, Paul Hamilton,  
Russel Waymire, Phong Nguyen, Peter Soliz

ORION International Technologies, Inc.

2201 Buena Vista Dr. SE, Suite 211

Albuquerque, NM 87106

## Abstract

*This paper describes a modeling, simulation, and integration framework developed at Sandia National Laboratories with support from ORION International Technologies, Inc. This framework, called Umbra, increases the efficiency for developing new applications and for integrating new and/or existing applications. A process for rapidly and effectively developing new modeling and simulation applications to reduce time and cost is discussed. This goal is achieved by using a time-tested object-oriented modular design, object-oriented design patterns and the implementation of a novel construct called World Modules. Umbra developers have embraced axiomatic design as a guiding principle. This paper describes the approach used to develop Umbra, lists representative projects, and describes specific examples illustrating how quickly and efficiently complex systems representations can be built in Umbra.*

**Keywords:** object-oriented, modeling, simulation, framework, scalability

## 1 Objective

The purpose of this paper is to describe a modeling and simulation integration technology, called Umbra, which has been developed and tested on numerous “wicked hard” problems. The goal of Umbra is to give the user timely, optimal results from simulation experiments of a wide variety of applications. Umbra provides a modeling and simulation

framework to users with a different vision of what makes a good tool or set of simulation tools. Our solution, presented herein, shows how we exploited a variety of technologies including modular design, object-oriented design patterns [1], and our novel construct called *World Modules* to develop a highly flexible and powerful environment for simulation and integration. As structured, Umbra facilitates the realization of many requirements related to modeling and simulation, interoperability between legacy models/ applications, hardware/human-in-the-loop, and other combinations. Umbra, designed originally to be a continuous time system, implements a hybrid system with continuous time, discrete event, and state driven capabilities.

## 2 Background

Umbra has been under development at Sandia National Laboratories, with support from ORION International Technologies, Inc. staff, since 1995. It was developed specifically because available simulation and integration environments could not satisfy Sandia’s needs in the areas of robotics development and systems integration. Prior to the development of Umbra, it was more difficult and complex to build a simulated system than it was to build a real system in the laboratory and test it in the field.

Instead of attempting to be an all-encompassing architecture for simulation and integration,

Umbra is a simple modular, composable framework that meets the requirement for both writing components from scratch and for leveraging existing external capabilities. Umbra uses design patterns [2] as a key building block and conforms to axiomatic design [3] by creating an independent, C++ based class called a module. Using Umbra modules as starting points, the complexity of the simulation is kept from growing geometrically because the modules do not require dependencies on other modules outside of their connectors. A module has output ports through which it presents data and input ports through which it reads data. Umbra implements a connector class for this behavior that acts as a go-between for passing data between modules. The only restriction for connectors is that they connect based on data types. The use of connectors separates inter-module dependencies while simultaneously increasing the number of functions for which a module can be used.

One of the major strengths of Umbra is its ability to define *Worlds* to handle data interaction types, such as radio components broadcasting messages, between different system structures [4]. The ability to have one *World Module* that efficiently processes, tracks, and manages interactions between  $n$  Children of that World limits the growth of complexity as the number of systems increase. The World Module idea comes from the factory design pattern and the associated Child Module is from the product design pattern. Using the radio example, if a World were not present, every radio would have to contain knowledge about the location of every other radio and every radio would have to calculate whether or not it can receive the message another radio is sending. With a World present, no radio has to know more than its own location and what message it is sending. The World performs all of the radio propagation calculations and passes the message along to only those radios that can detect it. Multiple Worlds are used to handle

different spectra within a simulation. Geometric visualization and collision detection are handled by two different Worlds. Different sensor types use different Worlds in combination with detectable properties [4], [5].

### 3 Methods

With Umbra, users will integrate and develop a new system simulation by following this process:

1. Define combinations of concepts, devices, behaviors, and events that describe the problem at hand.
2. Subdivide these combinations into component parts; e.g., sensors, behaviors, controllers, physics, and geometry.
3. For an initial starting point, find low-fidelity or ideal physics models as stand-ins for each component and ensure interactions between components are well-defined.
4. For each component part that requires more accurate models, find a similar existing model that lends itself to modification and can augment an existing model through inheritance. To achieve the most efficient implementation, these models will come from Umbra libraries, legacy systems, or commercial applications. Failing this approach, a new model must be created.
5. As model behaviors emerge, or unforeseen reactions between collections of components occur, it is important to determine the cause of the simulation artifacts. The component parts that caused most the simulation artifacts will require higher fidelity models with fewer approximations.
6. Modules or collections of modules representing these components or subsystems are replaced by the real world equivalents until the total system of systems collection operates according to pre-determined behavior.

*Note:* The above approach decomposes the evaluation of the simulation to the individual

components instead of the ensemble of components. This approach to systems integration based on validation of components is known as “emergent behaviors.”

### 3.1 Example 1

The following example simulation demonstrates elements of the above Umbra process. The simulation requires multiple vehicles to patrol the streets of a city with the task of detecting an entity or positional marker. The modules used for this simulation include:

- the vehicle entities
- behaviors that describe their motion
- a model of the city
- the motion of the entities to be detected
- the sensor to perform the detection
- the communications system to report positions or detections
- the set of high level behaviors to perform the general scenario.

The initial simulation uses low fidelity models whenever possible in order to have a representation of all components in minimum time. Inexperienced systems engineers may choose from the start of the development to focus on the design of the high-level behaviors to ensure realism. However, the high-fidelity component models may result in complex interactions between modules and display unpredictable behavior. This approach results in apparent failure in the high-level behaviors and makes the system more difficult to debug.

Usually the fidelity of the simulation is tailored to the parts of interest to the analyst developing the simulation. If the focus of the analyst is to design the search vehicle, the vehicle model may require modules for its suspension, mobility tables based on soil type, steering physics, accelerations, and visibility blind spots, but may not require high-fidelity communications or sensors modules. Conversely, if the analyst is trying to penetrate

a perimeter, the entities to be detected may require high-fidelity components such as camouflage, avoidance countermeasures, vulnerability to weapons fire, etc., while the surveillance system may require high-fidelity models of perception, cognition, detection, and object recognition modules.

Umbra provides the infrastructure that permits models to be generated from a baseline simulation by adopting simple, low-fidelity modules from a library and by giving high-fidelity behavior only to those components critical to the analysis and to the scenario, modifying only those modules required for proper simulation behavior.

From the initial runs, there will be indicators where additional fidelity is needed or in which areas additional fidelity will materially affect the outcome. To improve the fidelity of a given component, the component designer repeats the steps made to start the simulation. The designer creates stand-ins for each sub-component, and modifies the stand-ins until they behave appropriately. Once the infrastructure exists and passes the right information between components, the designer begins matching behaviors to equivalents in the real world, at least within an acceptable error tolerance.

There will always be uncertainties and assumptions within simulations. The objective is to make them as small or as low-impact as is reasonable and to not require an exact match between simulated entities and real ones. The goal of simulation is to test the feasibility of an idea. It is not usually intended to demonstrate that rivet X will fail at a specific time. Rather, it is intended to demonstrate that 15 vehicles patrolling a given city have between a 60-75% probability of catching an intruder or 30 vehicles have between an 80-90% chance of catching an intruder. If communication were not being considered a major factor and thus not accurately modeled, its effect on the

simulation may not be readily apparent and the derived solution may not be optimal.

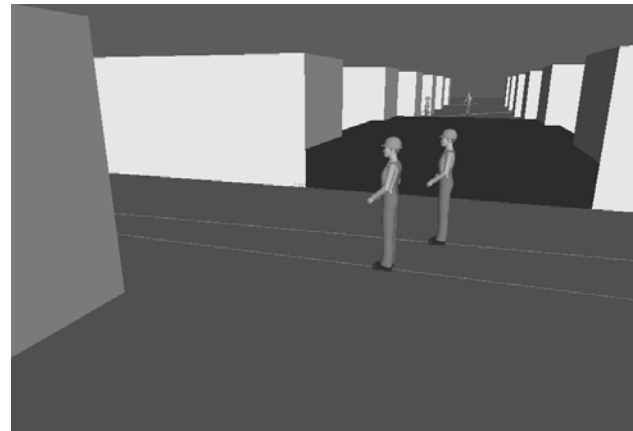
What the simulation might deliver, in this example, is the general trade-off between area coverage and communications density. Intuition would suggest that more vehicles and radios would result in more effective operations. Unfortunately, perfect communication is not generally achievable, so the simulation could include sources of these errors whenever using radios for communication. Ideal communications can be used as a stand-in for testing the idea, but, reliance upon ideal communications to prove a point is an assumption that is not always valid.

### 3.2 Example 2

Umbra has been employed in a diversity of projects ranging from relatively small scenarios running on a laptop computer to large, distributed projects running on multiple computers at multiple locations. The following scenario was generated in response to the Sago mine disaster in West Virginia in early 2006. The intent was to quickly demonstrate that Umbra could be effectively employed as a platform to evaluate technologies new to mining and to show how they might apply to mine safety and extraction of miners in the event of an accident (such as the one at Sago). The picture in Figure 1 looks simplistic but the underlying models and scenarios are complex.

A number of technologies have been singled out by mine safety inspectors and/or have been legislated as mandatory requirements for improved mine safety. One of these is radio frequency identification (RFID) and tracking. This technology has matured to the point where it is used routinely to track objects globally and RFID has been applied by a number of companies to track miners underground. Umbra provides a graphical means for displaying data from systems like the RFID TRACKER. The Umbra-based mine simulation system allows

mine owners and mine engineers to test this technology before the costly task of implementing RFID beacons. Through Umbra modeling and simulation, a programmer can model the effectiveness in tracking all miners with the optimal number of beacons and through modeling, optimize the placement of the beacons to achieve the greatest coverage at the lowest cost and complexity.



*Figure 1 – Section of a Room and Pillar Coal Mine with Evaluation of Mine Safety Technologies*

Another mine safety system that can be modeled is based on the idea of caching oxygen containers for emergency use. How many are needed, their location, their capacity, etc. are all variables that must be addressed. Umbra allows one to model different emergency situations with different scenarios (number of miners, location of rock fall or explosion, etc.) and test different cache strategies. The optimum cache strategy can be determined through stochastic processes derived from the Umbra model.

The miners are modeled as point locations. The goal is to construct targets that move about the mine at appropriate speeds at appropriate times in order to produce simulated TRACKER data. Functions include normal, scripted behavior that is a circulation between the face (area where coal is being removed) and the break areas. Emergency behavior include evacuation (seek and follow escape route), fire suppression (move toward fire if gas concentrations and

obstacles permit), and barricade (seal in place if gas concentrations are high and rising and escape was unsuccessful).

These examples are a sample of literally hundreds of applications ranging from robotic development/control and manufacturing to analysis of network-centric battlefield environments and the development of cognitive systems. Umbra is a powerful and flexible development and integration environment. It has been used for an array of applications within Sandia National Laboratories, the Department of Defense, the Department of Energy, and other federal government agencies. ORION has used Umbra to integrate legacy code, commercial off-the-shelf software, and other programs as part of a complex environment such as the National Aeronautics and Space Administration's Small Aircraft Transportations System (SATS) [6]. Umbra was used by ORION to implement a retinal surgery training system [7].

The capability to employ Umbra as an integration platform has been demonstrated repeatedly using a wide variety of methodologies. With the source code or a well-defined Application Program Interface (API) for an application (or legacy model), it is standard practice to encapsulate the code (using the object oriented paradigm) and make these programs into Umbra modules – to be instanced using the standard scripting processes. Among the first approaches to general integration to be studied was the Defense Modeling and Simulation Office High Level Architecture (HLA) federation between simulation environments. A highly successful and efficient HLA implementation (an Umbra World Module) was created for the Joint Virtual Battlespace Program led by the US Army Research and Development Engineering Command.

## 4 Acknowledgements

The authors wish to acknowledge the developers from the Sandia National Laboratories' Intelligent Systems and Robotics Center. ORION staff supported the development of Umbra and now has licensed the software from Sandia for commercial purposes. This publication incorporates technology licensed from Sandia National Laboratories under Sandia License Number 02-C01059.

## 5 References

- [1] Shalloway, Allen, *Design Patterns Explained: A New Perspective on Object-Oriented Design*, 2<sup>nd</sup> ed. (Boston, MA: Addison-Wesley, 2004).
- [2] Alexander, C., S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King and S. Angel, *A Pattern Language* (New York: Oxford University Press, 1977). [Review by Nikos Salingaros and Stewart Brand].
- [3] Suh, Nam P., *Axiomatic Design: Advances and Applications* (New York, NY: Oxford University Press, 2001).
- [4] Gottlieb, Eric, et al., "The Umbra Simulation Framework as Applied to Building HLA Federates," *Proceedings of the 2002 Winter Simulation Conference*, E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.
- [5] Hart, Brian private communication, Sandia National Laboratories, Intelligent Systems and Robotics Center, March 1, 2005.
- [6] Hamilton, Paul, et al., "Rapid Development of a Multi-Aircraft Aviation System Simulation," *Proceedings of the WORLDCOMP 2006 - International Conference on Modeling, Simulation, and Visualization Methods*, June 26-29, 2006.
- [7] Barriga, Eduardo, et al., "A Training System for Photodynamic Therapy Using Modeling and Simulation," *CBMS 2006 – The 19<sup>th</sup> IEEE International Symposium on Computer-Based Medical Systems*, June 22-23, 2006.