2004-06

Connecting Simkit, CombatXXI, and Naval Simulation System (NSS) via XMSF Web Services

Blais, Curtis L.
Connecting Simkit, Combat\textsuperscript{XXI} and Naval Simulation System (NSS) via XMSF Web Services

Curtis Blais  
NPS MOVES Institute

John Ruck  
ROLANDS & ASSOCIATES Corporation

22 June 2004
Background

• Sponsor: Director, Assessment Division, Office of the Chief of Naval Operations (OPNAV N81)

• Goals:
  ▪ Apply Extensible Modeling and Simulation Framework (XMSF) concepts to analysis tools
  ▪ Connecting Simkit, Combat XXI and NSS via XMSF Web Services
  ▪ Establish transformational analytical framework
    ▪ From: monolithic, closed system designs
    ▪ To: M&S frameworks that are open and permit modular, loosely coupled components to be rapidly integrated to create agile analytical capabilities
Schedule Milestones

• January – May 2004
  ▪ Establish software teams, build capabilities
  ▪ Begin hiring analysis team to design models
• June: MORS Symposium, Monterey
  ▪ Show current progress, NSS/Simkit/CombatXXI framework for conducting analysis tasks
• July: Continued tool development and analysis demonstration
• September: project + analysis briefings
  ▪ OPNAV N81, N70, N60, CNO staff, others
• December 2004: final reports

Fast track throughout
Key Technologies: XMSF

Extensible Modeling & Simulation Framework

- A composable set of standards, profiles, and recommended practices for web-based M&S
- Foundational precepts: Internet technologies, Extensible Markup Language (XML)-based languages, and service-oriented architectures (e.g., Web services) for simple messaging
- Enable a new generation of distributed M&S applications to emerge, develop, interoperate

http://www.movesinstitute.org/xmsf
Technology Task

Analytical Framework: WCM-14

- Integrate Naval Simulation System (NSS) with Simkit Discrete Event Simulation (DES), using XML messaging via Web services
- Interconnect CombatXXI with NSS events
- Small-scale demonstration of integrated capabilities (SOF scenario)
- Fundamental technical connectivity is now demonstrated via XML, shared-event queue
Simkit

- A Discrete Event Simulation Application Programmers Interface (API) in Java
- Developed at NPS
- Open source (GNU GPL)
- GUI/tool-kit under development (Viskit)
- Not a model, a model “engine” - schedules and processes events
Naval Simulation System (NSS)

- Widely used/understood analytic tool
- NAVAIR program manager, Metron prime, 10 years development
- Theater-level modeling of vehicle platforms, sensors and weapons
- Written in C++
Combat XXI

- Next-generation analytic tool for modeling theater-level ground warfare Army, Marine Corps
- New software under active development
- Government owned
- Already embeds NPS Simkit code
Two Main Tasks

• Develop the technical capability
  ▪ Interconnect NSS and Combat\textsuperscript{xxi} at the event level
  ▪ Use Web Services/XML/SOAP/XMSF
  ▪ Develop reusable methodology

• Conduct demonstration of analysis capability
  ▪ Tacsits of interest in ongoing OPNAV PR-07 Campaign Analysis
Technical Approach

• Development Philosophy
  - Rapid prototyping/Incremental development
  - Open/web-based standards
    - XML-RPC/SOAP/AXIS/Tomcat
  - Goal is not to improve existing simulations
  - Goal is to develop a methodology to allow the best parts of existing simulations to be used
Web Services Standards

How a client talks to a Service provider depends on the WSDL

Web services provider – uses UDDI to register a Web service within the repository

Web services client – uses UDDI to find an appropriate Web service

Web services registry

UDDI

WSDL

XML/SOAP

XML/SOAP

XML/SOAP

XML/SOAP

Web

UDDI

UDDI

Green pages

White pages

Yellow pages

UDDI

WSDL

SOAP

FTP

HTTP

SMTP
Simkit as a Web Service

• Single event queue for all simulations
• Simulations loosely coupled using event listener pattern.
  ▪ One simulation can register to “hear” (and therefore process) events scheduled by the other simulation
  ▪ Requires translation of event data to a common semantic in addition to syntax.
Development Stages

- Modified NSS to use Simkit as its model engine (via Java Native Interface)
  - Minimal modifications of existing NSS code to re-direct simulation engine calls to Simkit
- Implemented a simple Simkit simulation to listen for NSS weapon impact events.
- Added the ability for a Simkit entity to inject a contact report into NSS.
- Re-engineered as Web Services.
Three Web Services

• SimkitWS
  - Simkit wrapped as a Web Service so simulations can schedule, cancel, and receive events.

• NssWS
  - NSS wrapped as a Web Service

• NativeSimkitWS
  - A wrapper to allow any Simkit simulation to interact with the SimkitWS
  - Goal is “main method compatibility”
Design Patterns/Concepts

• Data proxies and data translators make data from one model available to the composite

• Registered events are those events that will be exposed. (n of ~200 NSS events)
  ▪ Non-registered events are wrapped in a default event and would not make sense to another simulation.

• The 3 Web Services handle the incoming SOAP method calls; Proxies handle the outgoing (Client) method calls.
Contact Report generated in Simkit simulation heard by NSS

NativeSimkitWS

SimkitWS

TestSpotter

Run

SendContact

SimEventFilter

NssWS

NSS MessageManager (on a “surrogate host”)

NssWS

Nss_simkit_proxy

TestSpotterAdapter

SendContact

MessageManagerProxy

MessageManagerReverseProxy

Contact Report generated in Simkit simulation heard by NSS

NativeSimkitWS

SimkitWS

TestSpotter

Run

SendContact

SimEventFilter

NssWS

NSS MessageManager (on a “surrogate host”)
Weapon Impact from NSS Heard by a Simkit Simulation

NssWS

NSS (Event_manager_proxy)

ReachTarget

TargetNotFound

(NSS)

Fire

SimkitWS

SimEventFilter

NativeSimkitWS

SimulationWS (NativeSimkitWS)

WeaponImpactListener

ReachTarget

TargetNotFound
“Hello World” Scenario

SOF (in Simkit simulation) detects target
SOF sends contact report to ship (in NSS)
Ship fires at contact report
Umpire (in Simkit simulation) hears weapon impact
Analysis Demonstration

- Using the composite system, demonstrate the ability to conduct analysis.
- Final piece of proving the concept
- Shows it’s not just a “science project”
• WCM-14 Examine Special Operations Forces in Surveillance and Reconnaissance Role
• WCM-19 Examine Special Operations Forces in Direct Action Role
• WCM-15 Examine the raid capability of smaller, task-organized elements of the Marine Expeditionary Unit (Special Operations Capable). Examine also the distributed fires capability provided by emerging “missiles-in-a-box” systems.
Next Steps

• Continued Software Development
  ▪ Enable additional NSS events
  ▪ Integrate Combat\textsuperscript{XXI}
    ▪ If all goes well, it should drop in for initial functionality

• Analysis
  ▪ Gain Combat\textsuperscript{XXI} scenario development expertise
  ▪ Identify needed software changes
  ▪ Composite scenario implementation
  ▪ Design of experiments
  ▪ Conduct and analyze model runs
Questions?

Curtis Blais
NPS MOVES Institute
831-656-3215
clblais@nps.edu

John Ruck
ROLANDS & ASSOCIATES Corporation
831-656-1006
jlruck@nps.edu or nss@rolands.com
Related Briefs

• A Transformation Framework for Design, Development, and Integration of Simulation Models (WG-11/31 Wed 1330-1500)

• Simkit Analysis Workbench for Rapid Construction of Modeling and Simulations Components (WG-26 Thurs 0830-1000)
Backup Slides
Standards

• XML-RPC: Remote procedure call based on encoding as XML and using HTTP as the transport. Available in almost any language. (Implemented as Apache XML-RPC)
• SOAP: Grew out of XML-RPC
• AXIS: The Apache SOAP implementation. Supports SOAP, WSDL, and JAX-RPC.
Challenges and Lessons Learned

- **Visual C++ Linker errors when moving from Intel/W2K to AMD/XP**
- **Static data in a Web Service**
  - Runs in the same JVM as Tomcat, so classes and dynamic libraries are not reloaded between runs. (In fact a “run” is just a series of unrelated method calls as far as a Web Service is concerned.)
    - Even the NSS dynamic library is not reloaded between runs.
  - Currently have to restart Tomcat between runs.
  - Modifications to Simkit to convert from static
  - Need a solution for existing applications
Challenges and Lessons Learned

- **Axis for C++ issues**
  - XML encoding of Objects not compatible with Java Axis
  - Mapping of C++ data types to SOAP data types questionable
- **Web Services implementation currently 2.5 times slower than non-Web Services**
  - Need to look into alternate transport implementations, the default is HTTP which has the overhead of making and breaking the connection for each XML-RPC call.
- **Timeouts during XML-RPC calls**
  - Default behavior is for the call to timeout after 60 secs.
  - Had to set the timeout to never.
  - An Axis extension to the XML-RPC spec.
- **Incompatibility of different XML-RPC/SOAP implementations**
  - Compatible at the SOAP message level, but not at the code level
  - Can’t abandon AXIS for another implementation without rewriting client side code.
### Projects Overview

#### Analytical Modeling Framework

<table>
<thead>
<tr>
<th>NSS</th>
<th>Simkit Combat&lt;sup&gt;XXI&lt;/sup&gt;</th>
<th>Simkit Analytical Workbench</th>
<th>JFEO Modeling</th>
<th>Improve Strike Module</th>
<th>with AT/FP prototype</th>
</tr>
</thead>
</table>

- **XMSF / X3D vis / Web Services**
- **N81 Analyses**
- **SAVAGE and MAS**

**AT/FP**: Anti-Terrorism / Force Protection

**Combat<sup>XXI</sup>**: Next-generation Army/Marine analytical model

**JFEO**: Joint Forcible Entry Options

**MAS**: Multi-Agent Systems

**NSS**: Naval Simulation System

**SAVAGE**: Scenario Authoring and Visualization for Advanced Graphical Environments

**Simkit**: Discrete Event Simulation (DES) Application Programmers Interface (API)

**X3D**: Extensible 3D Graphics

**XMSF**: Extensible Modeling and Simulation Framework
Analysis Timeline

WCM-19

WCM-14

Pre-Hostilities

• Assembly and Force Closure
• Intelligence and Operational Preparation of the Battlespace

WCM-15

Seize the Initiative

• Gain air, sea, space, and information superiority
• Establish Sea Control in area and lines of approach
• Deter and neutralize strategic targets

Decisive Operations

Post-Hostilities

Note: Arbitrary, conceptual representation
WCM-14 Problem Definition

Examine Special Operations Forces (SOF) in Surveillance and Reconnaissance (S & R) role

• Compare the operational efficacy and costs of SOF to an alternative future system of airborne Intelligence, Surveillance, and Reconnaissance (ISR) assets
  ▪ SEALS versus UAV-based architecture (Global Hawk, Predator, J-UCAS)
  ▪ Intelligence and Operational Preparation of the Battlespace (IPB/OPB)

• Measures of Effectiveness
  ▪ Responsiveness to Commanders’ changing requirements
  ▪ Operational costs of delivering and sustaining capability
    ▪ Bandwidth
    ▪ Assets to deliver, sustain continuous coverage
  ▪ Risk to platforms, personnel, campaign objectives
WCM-14
Surveillance & Reconnaissance
• SSGN deliver SOF
• SOF ingress to target area
• Coordinate with sea-based commanders
• Provide surveillance and reconnaissance
Examine Special Operations Forces in Direct Action (DA) role

• Compare the capability of SOF targeting to that of a centrally-located targeting facility
  ▪ SEALS versus imagery interpretation and aimpoint mensuration at the Combined Air Operations Center’s (CAOC)
  ▪ SOF Direct Action
    ▪ Organic neutralization capability
    ▪ Forward Fire Support Observers/Controllers with strike assets in direct support
  ▪ Maritime Component Commander’s priority targets
    ▪ Enemy’s maritime access denial capability: coastal Anti-Ship Cruise Missiles (ASCM), mines, submarines

• Measures of Effectiveness
  ▪ Targeting accuracy, processing requirements, and latency
  ▪ Time to achieve required target attrition
  ▪ Operational costs of delivering and sustaining capability
    ▪ Bandwidth
    ▪ Assets to deliver, sustain continuous coverage
  ▪ Risk to platforms, personnel
Direct Action
- SOF neutralizes targets
- SOF controls supporting fires
- Coordinate with sea-based commanders
Examine the raid capability of smaller, task-organized elements of the Marine Expeditionary Unit (Special Operations Capable). Examine also the distributed fires capability provided by emerging “missiles-in-a-box” systems.

- Compare the strike effectiveness of raids and distributed fires systems to that of the Expeditionary Strike Force’s strike capability
  - SOF-like employment of MEU(SOC) elements versus Tomahawk and Carrier Air Wing strike capability.
  - Rapid deployment forward of distributed capability for indirect fires
  - Commander, Joint Task Force and Combatant Commander’s priority targets
    - Ballistic Missile Facilities, Integrated Air Defense System, Weapons of Mass Destruction, etc.

- Measures of Effectiveness
  - Responsiveness to commanders’ changing requirements
  - Time to achieve required target attrition
  - Operational costs of delivering and sustaining capability
    - Bandwidth
    - Assets to deliver, sustain continuous coverage
  - Risk to platforms, personnel
Advance Force Operations

WCM-15
Joint Forcible Entry Operations
• Raids by SOF and MEU(SOC)
• Distributed fires capability provided by “missile-in-a-box” type systems
Web Services Concept

Network Environment

XML Messages

Web Services Interface

Back-End Systems

DBMS

.NET

J2EE

CORBA

Adapter

ERP