Modular Interoperable Synthetic Environment: Final Report

1-1-2001

Brian L. Pettitt

Find similar works at: http://stars.library.ucf.edu/istlibrary

University of Central Florida Libraries http://library.ucf.edu

Recommended Citation


This Research Report is brought to you for free and open access by the Digital Collections at STARS. It has been accepted for inclusion in Institute for Simulation and Training by an authorized administrator of STARS. For more information, please contact lee.dotson@ucf.edu.
Modular Interoperable Synthetic Environment

December 13, 2001

Brian L. Pettitt

Institute for Simulation and Training
University of Central Florida
Final Report

“Modular Interoperable Synthetic Environment”

Prepared by
Institute for Simulation and Training
University of Central Florida
3280 Progress Drive
Orlando, FL 32826-0544

Prepared for
U.S. Army Simulation, Training, and Instrumentation Command
12350 Research Parkway
Orlando, FL 32826-3276

IST-CR-01-23

Contract N61339-01-C-0036

Unclassified

For additional information please contact:
Mr. Brian L. Pettitt 407-882-1435 bpettitt@ist.ucf.edu
1. Project Overview

Many current Modeling and Simulation (M&S) applications are of monolithic design, not easily decomposed, and resistant to improvement and reconfiguration. Interoperability among these systems, which incorporate varying levels of fidelity and diverse communications protocols in the absence of a definitive and comprehensive set of standards, is also a critical issue that has yet to be fully addressed. The Modular Interoperable Synthetic Environment (ModISE) framework development project is an attempt to research and address some of these hurdles toward the advancement of the state-of-the-art in M&S technology. The architecture is based upon two main concepts, which are interoperability and composability. The initial targeted use for the framework is to function as a M&S concepts testbed for the Army Simulation Training and Instrumentation Command (STRICOM).

The ModISE can be used to facilitate interoperability across heterogeneous data representations (HLA, DIS, streaming voice/video, internet, custom protocols, etc...) and also provide composable functionality to externally connected environments or clients. Users compose an execution environment, via a GUI, from reusable components that are stored in a repository. At runtime, the selected components are downloaded from the repository, configured and initialized inside an instance of a ModISE Interoperability Engine (IE), and used to provide augmented capability or to facilitate interoperability across one or more externally connected clients. A particular instance of the ModISE IE can be composed as a functioning federate, a middleware interface, an inter-federation gateway, or for a variety of other purposes.

2. Project Status

2.1. Technical Work

- Definitions:
  Nlp = natural language processor  
  CI = context interpreter  
  VoiceModISE = class that interfaces between HLA and oaa  
  VoiceMTM = class that resides inside ModISE  
  Hla2Oaa = software interface between ModISE and Oaa  
  Vfed = a stripped down version of VoiceFed without voice recognition which sends voice strings to the HLA federation.  
  VoiceFed = complete Voice Recognition federate  
  IE = interoperability engine  
  FSB = finite state machine  
  MCV = ModISE composed vehicle  
  Dragon = voice recognition engine  
  MDI = ModISE dismounted infantry  
  Espec = execution specification
• We spent part of our time learning about the Java tools used by SAIC in the ModISE project so that we can use it in our development as well. These include jmake (make for Java), jdeps (dependency creator for java), and bash shell.

• We also spent time doing some coding for creating a vehicle composition MTM (VehCompMTM) which can merge attributes from 2 external vehicles, which was IST's proposed scheme of vehicle composition.

• We attended the ModISE IPT meetings on 1-19-01 and follow-on meeting on 1-24-01 to discuss work to be done in the next phase of the project.

• Due to STRICOM's budget constraints, it has been decided to combine resources from IST and SAIC to work on SAIC's proposed scheme of vehicle composition. We are going to re-focus our effort to the new scheme of vehicle composition.

• The Turtle Beach sound card that came with my pc ordered in November had a problem. We have tried to get it to work but have been unsuccessful so far. The reason being that Dell does not have the driver yet for the card under Win2000. The integrated sound on the motherboard works well.

• We attended ModISE IPT meetings on February 5th and 21st to discuss and plan for the next ModISE demonstration. Our involvement for the next demonstration is to use voice commands to direct the ModISE composed vehicle (MCV) to move on the terrain. Specific voice command phrases are yet to be determined.

• To ensure that VoiceFed, voiceMTM, DISAF and components were integrated correctly we ran the programs using the integrated sound on our PC, a problem with Dragon was found and solved.

• We spent some time learning how to compile and rebuild VoiceFed using Microsoft Speech SDK and Dragon Speech SDK. The re-compiled version was tested to make sure it worked with ModISE.

• ModISE is using Java language for development and Kawa is recommended as the Java debugger to use. While waiting for the ModISE funding to come through we downloaded a trial version of Kawa from its web site and installed for evaluation and familiarization.

• IST attended ModISE IPT meetings on March 5th, 16th and 23rd.

• IST's involvement for the next demo is to use voice commands to direct the ModISE composed vehicle (MCV) to move on the terrain. Specific voice command phrases are: "MCV set speed <number>" where <number> is 0-150, "MCV set heading <number>" where <number> is 0-359.
• We installed VoiceFed, Dragon and other software on the Stricom lab computers for next demo.

• A pre-integration test was done on 3-15 and the final integration test for the demo was done on 3-30. SAIC was running modise to compose an MCV vehicle and then use VoiceFed to issue the two voice commands to change the speed and heading. A stealth display was used to view the MCV. The MCV responded to the commands correctly.

• SAIC wanted to use a large number for the set speed command. VoiceFed has problem with number longer than 3 digits. A longer number was split into two numbers with 0’s added to the end. After talking to Richard Urich (VoiceFed project member), and after studying the source code, the cause of the problem was found. Numbers in VoiceFed were originally used exclusively for coordinates. So a long number is always split into 2 numbers for x and y coordinates with 0’s padded at the end to make each number 5 digits in length. Since modise needs to handle numbers for speed and heading, a long number should be treated as one number and no splitting should take place. The fix has been made and tested. ModISE is using Java language for development and Kawa is recommended as the Java debugger to use. Kawa was ordered and received and installed on one of IST’s PC. We spent some time to learn how to use it.

• We received a list of voice commands from SAIC that VoiceFed is going to support for the demo on 4-25. These voice commands are to direct the MCV and MDI to move, change speed setting and directions. We trained VoiceFed to recognize these voice commands and send them as text strings to the VoiceDemo federation execution so that the entities can respond appropriately. We had our first integration test on 4-20, dry run on 4-24 and the actual demo on 4-25. The final demo was completed successfully. Originally, the plan for our next task was to develop the whiteboard display software to display state changes for the MCV in the IE. After talking to Mark Biddle, our task was redirected to work on MCV Ordnance MTM which is more critical for ModISE. We have talked to Chris Otto of SAIC to learn about the MCV aggregation protocol which is used by the various MCV components to aggregate with each other. We have begun to work on the Ordnance MTM. The essential part of the task is to model the flight path of the munition after the launch and determine if it results in a hit or a miss at the end of the flight. Its coordinates needs to be broadcasted continuously to the federation during the flight. There are at least two different coordinate systems that we need to contend with (UTM and GCC). The modeling is going to be done in UTM and the contents of the weapon fire and detonation interactions will be expressed in GCC. Conversion between them will be developed. After getting through the conceptual design, we will begin my program design and coding.
We have learned how to interface with the terrainDatabaseAccessMTM and weaponfire interaction coming from a firing entity and also learned how to use the common ModlSE classes that are needed by OrdnanceMTM. With conceptual design for modeling the munition flight completed, we have started the coding phase for the OrdnanceMTM.

We have trained the voiceFed to recognize a few voice commands requested by Ted (SAIC) on 5-21-01.

We attended ModlSE IPT on May 24 and several ModlSE interim meetings in May.

On May 24, STRICOM has instructed all the ModlSE contractors to re-focus their effort to integrate Websim3D and DISAF together to allow them to interoperate via ModlSE. The goal is to demonstrate ModlSE's interoperability capability and the high level GUI used for constructing espec for ModlSE execution. We are shooting for the demo around the end of June.

We are currently focused on writing the DISAF proxy MTM for the Websim3D-DISAF interoperability demo.

Received a new computer for ModlSE project. Have asked info services to configure and install the standard IST S/W on the computer.

For the month of June, all the ModlSE contractors including IST were focusing on integrating Websim3D and DISAF together to allow them to interoperate via ModlSE.

Seng Tan has completed the coding and testing for DISAF proxy MTM. This MTM, together with Websim3D proxy MTM and RPRFOM converters developed by SAIC, allow DISAF and Websim3D to interoperate via the ModlSE Interoperability Engine.

Integration tests were performed before the final demo that took place on June 28.

For the demo, several DISAF entities were created and the data were brought in to a data port on the IE via a gateway. Websim3D entities were also created and the data were brought in through another data port on the IE. The data were channeled through the Proxy MTM's and FOM converters (which convert between RPR0.5 used by websim3D and RPR1.0 used by gateway). With the use of the ModlSE, DISAF entities were able to appear on Websim3D display screen and Websim3D entities were able to appear on DISAF display. They were able to interact with each other through weapon fire.

IST visited the ModlSE playground and SAIC demonstrated the system. IST is reviewing the documentation and is in the process of clarifying the startup instructions for users.

IST briefing by Dr. Ron Hofer on JMASS Discussion centered on JMASS Program structure and JMASS components.

JMOOSE (Joint Modeling Object Oriented Synthetic Environment)
Navy requirement to incorporate SEDRIS (Synthetic Environment Data Representation & Interchange Specification)
SCORM (Shareable Content Object Reference Model)

Meta-tag structure/approach to ADL (Advanced Distributed Learning)
• Developed a basic understanding of ModISE infrastructure and how to make an MTM (using Java). Throughout the month there were several meetings with SAIC to become familiar with the environment at the TDC.
• The ModISE environment was setup on a computer here at IST.
• The documentation for ModISE was centralized for access from a single web site. This included revision of the playground start-up procedures and the addition of a FAQ. The work done will continue to be an aid as the development of ModISE continues because as new features are added, tutorials and instructions can easily be added to this web site.

• ModISE Integrated Process Team Meeting Agenda/Items – 29 August 2001
  o JMASS (Joint Modeling and Simulation Systems) meeting was briefed by Connie Perry. JMASS and ModISE requirements need to focus on Real-Time Simulation
  o ModISE Playground status
  o CTPS integration into ModISE status
  o IST team final status