

# Visualization of Stimulated DODAF Architectures

Johnny Garcia  
SimIS, Inc.  
200 high Street suite 402  
Portsmouth, VA 23704  
[Johnny@simistech.com](mailto:Johnny@simistech.com)

## Keywords:

**DODAF, Executable Architectures, Discrete Event System Specification (DEVS),  
Core Architecture Data Model (CADM), Open Standards**

*The ability to graphically represent the dynamic stimulation of static DODAF artifacts using existing tools is one of the goals of the Executable Architecture Analysis Modeling (EAAM) project for Science and Technology research funded by the DOD. The Net-Centric integrated architecture products modeled using existing COTS tools are stimulated via the EAAM environment to give user specified measures of performance based on command designed static DODAF architectures. Different technologies that may be used to visualize the stimulation of architecture products is the purpose of this paper. Some of the Technology used is collaborative software, portlet programming and Java API's for JFrames which will web enable EAAM and make the stimulated architecture artifacts viewable in a format that is easier to interpret and understand. The input into the EAAM environment is data from combat models in various formats which are fed into the architecture products, via an EAAM web service. The user will be able to see the simulated stimulation of the architecture products as they are happening and get real-time feedback from a specific mission thread chosen by the user. The visualization should be scalable so that you can zoom in and out of particular views. Selectable so you can choose the views you want to view, and secure so that only those with a need to know would have access to the information. Issues of concern are the synchronous stimulation and viewing of separate views on one parent page. Another issue is adaptive user views which would allow the user to specify the best layout for their particular needs.*

“Preattentive processing is a human’s ability to rapidly and accurately analyze visual properties of images at a very low level, without requiring focused attention; it is the fundamental factor in the effectiveness of visualization systems”(Conti 4).

## **Background**

Executable Architecture Analysis Modeling known as the EAAM project is the catalyst to the visualization of stimulated Department of Defense Architecture Framework (DODAF) compliant artifacts. The EAAM project prototype concentrated on integrated architecture models that focused on a time sensitive targeting thread. A combat scenario was created using Joint Semi Automated Forces (JSAF), which communicated through a High Level Architecture (HLA) runtime infrastructure. Federates publish and

subscribe attribute data of specific entities created in JSAF (Kulh 53). Events, States, and Message passing form the core communication between the combat model, simulation engine and the visualization of stimulated DODAF architecture products. The inputs that the EAAM environment can process are in standardized data formats such as Core Architecture Data Model (CADM), exported from Provision (Metastorm), and XML. Being able to input JIMES and TENA data is still being researched at this time. The software, and methods used to animate or display the stimulated architecture products will be discussed in the sections that follow. Section 1 will explore the commercial off the shelf (COTS) tools used. Section 2 will discuss the programming language and development environments explored, and Section 3 will discuss the user interface.

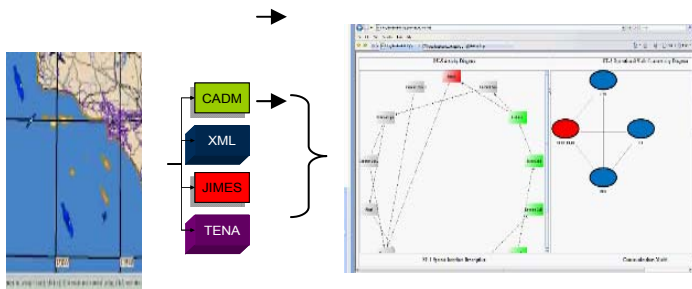
### **Executable Architectures and the EAAM project**

The Department of Defense is moving towards net-centric battle spaces and the COTS tools as well as the development environment that EAAM experimented with for the prototype needed to make the executable architectures viewable using current Internet technologies and protocols. JSAF 2007 is a government off the shelf (GOTS) tool that EAAM used to create the combat scenario. The visualization of the combat scenario can be viewed in a 2D battle space or a 3D Modstealth / Jstealth view. MetaStorm's ProVisison was used to create static DODAF architecture data and the java based software from XJ Technologies called AnyLogic was used to bring them to life (AnyLogic). Initially AnyLogic 6.2 Advanced was used as the simulation engine for the executable architectures and the ability to animate model components or active objects was built in. AnyLogic allows 2D and 3D views of model animation. The visualization was then saved as an applet then published in SharePoint. The members of the EAAM project could then login to the site via the Internet. Tools such as SharePoint by Microsoft, which is available with their server load, provide a collaborative environment to visualize EAAM's stimulated architecture products. SharePoint was used because it is easily secured, only those authorized to view the sites would be able to. SharePoint's collaborative capabilities were also a plus because it enables multiple authorized users to view the same information simultaneously. A site administrator has the ability to add and delete web parts so different users would be able to have different views of the same information. The fact that SharePoint inter-operates with different Microsoft products such as PowerPoint, Outlook and other business products helps with the overall accessibility of information.

Microsoft security groups, domain permissions, as well as the SharePoint site administrator actions all come together to provide a secure platform to view the applet. The communication between the combat model, simulation engine, and the visualization was not synchronized using the AnyLogic applet, however, the AnyLogic applet was saved with a current experiment and viewable in SharePoint by opening up the applet in a web part and manually launching the saved experiment. The visualization of specific scenarios could be launched via SharePoint manually using an AnyLogic applet that was created initially on the desktop then uploaded into SharePoint. The applet was only stimulated through Anylogic and not from the combat model so the input of information was an issue that will need further research. Initially the executable architectures were

graphically represented by COTS modeling tools, but the goal was to visually represent the DODAF models in a way that would be shareable and viewable through the web using open source software. Combat scenarios and the visualization of the DODAF products is stimulated and controlled by the interactions between the combat model and the simulation engine so timing is a major concern. The fidelity of the end to end graphic representation of executable architectures must be held to a high standard in order for the users to benefit from the EAAM capability. The COTS tools explored have made that possible.

Combat model → Data generation → EAAM environment → Visualization



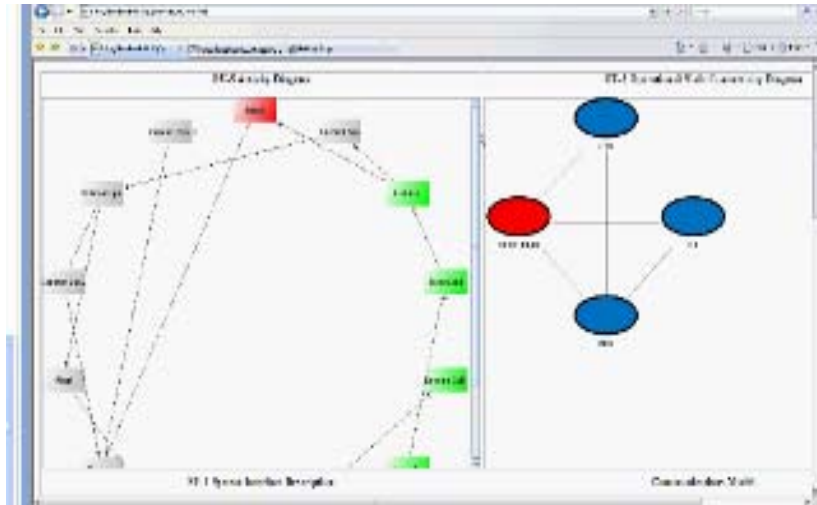
**Figure 1 shows the flow of information in the EAAM process**

### Technical Approach

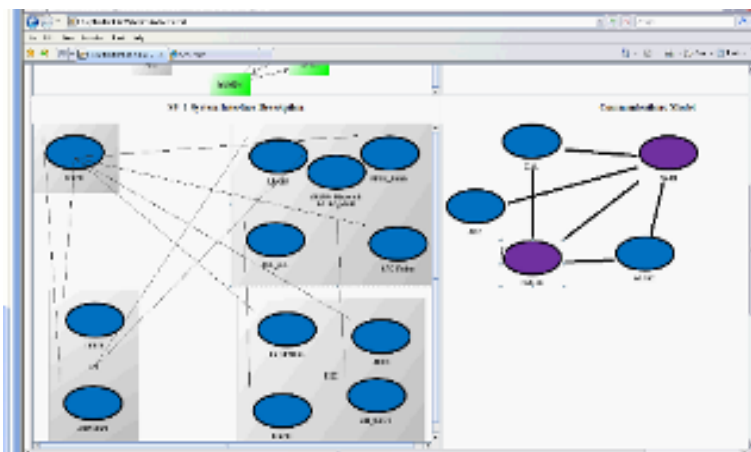
The visualization uses an Apache web server and runs java applets that communicate back and forth with the combat model to execute the processes that pertain to the mission thread that was selected for the simulation.

Java is the language used for most aspects of the EAAM visualization.

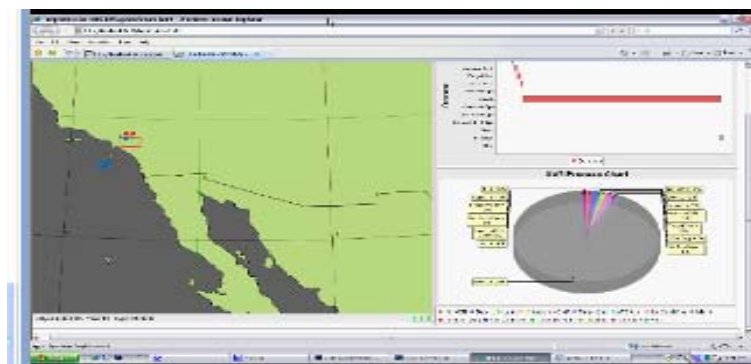
Separate Java applets were used for each architecture product within a frame allowing the user to view individual architecture products (Figure 2, 3, and 4). C# was looked at for user controls in SharePoint.



**Figure 2 shows the OV-5 and OV-2 operational views**



**Figure 3 shows the SV-1 system view and communications model**



**Figure 4 shows the OpenMap 2D viewer and Analysis data in a sample format**

The applets are dynamic and change colors according to the function being performed. The visualization of some of the integrated architecture is similar to an undirected graph

using oval and rectangular icons to represent entities connected by edges which represent links. The different colors represent the different states of the entities to the user.

**Table 1 list of colors and their functions**

<b>Color / View</b>	<b>Function</b>
<b>Red</b> / OV-5 / OV-2	<b>Current Activity</b>
<b>Green</b> / OV-5	<b>Activity has been executed</b>
<b>Yellow</b> / OV-2	<b>Waiting</b>
<b>Light Blue</b> / Comms model	<b>Voice</b>
<b>Purple</b> / Comms model	<b>TENA transmission</b>

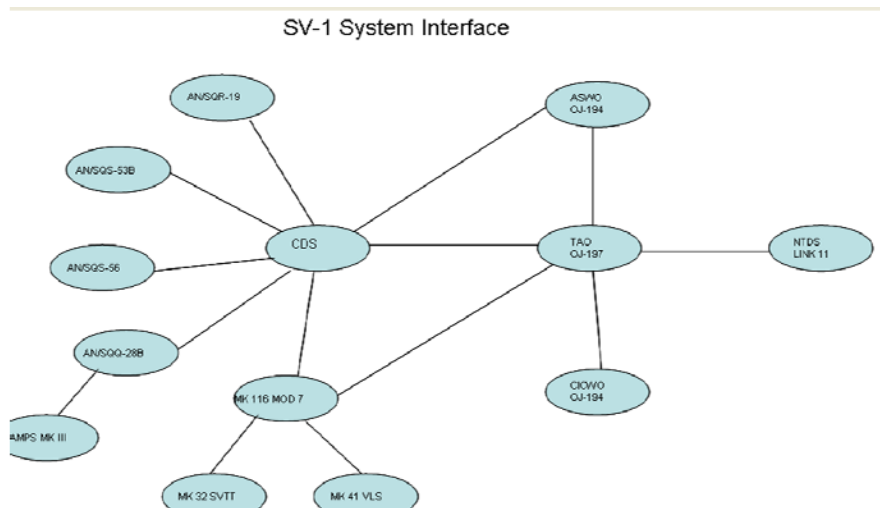
### **Discrete Event System Specification (DEVS) formalism**

The visualization of the simulated architecture should be deployed in one or more distributed environments such as server/ client, collaborative, and or a centralized location such as DARS. The DEVS formalism is capable of handling hierarchically structured systems and supports coupling and composability of atomic models. DEVS is well adapted to timed simulation and is capable of generating abundant quantitative data via simulation.

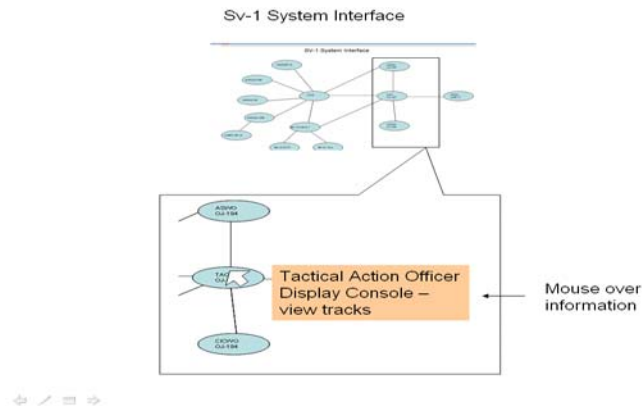
### **Scalability, Selectability, and Adaptability**

Scalability, Selectability, and Adaptability are the characteristic that need to be addressed in the user interface because of the vast amount of information that can potentially be available with the EAAM capability. Hundreds of architecture products could be viewable but not every user will want or need to view every integrated DODAF artifact. The user interface (Figure 5) should allow the user the option to upload the data in DOD approved data formats. The uploaded data would have to be verified as in the case of XML data that would have to be parsed to verify the data was in an acceptable format. Feedback to the user for common errors should be available upon rejection of uploaded data. Once the user test plan and uploaded data are validated, the stimulation of the architecture will be a click away. Applet views will have to be scalable so that they can be resized for better viewing specific to the user. The views should also be adaptable so that every time a user logs in their most frequently viewed DODAF products could be available without the user having to select the views they want every time they login. The option to change views should always be available. The structure of the views is variable depending on the type of information to be displayed for example the operational view (OV-4) organizational model is viewed using hierarchical data. The top organization is the root and the subordinate organizations are viewed in a tree structure. Another example is the operational view Activity Modeler OV-5 (Figure 2) which is created with rectangles that represent activities and the activities behavior is represented by different colors. A high level view of all the net-centric integrated architectures should be available

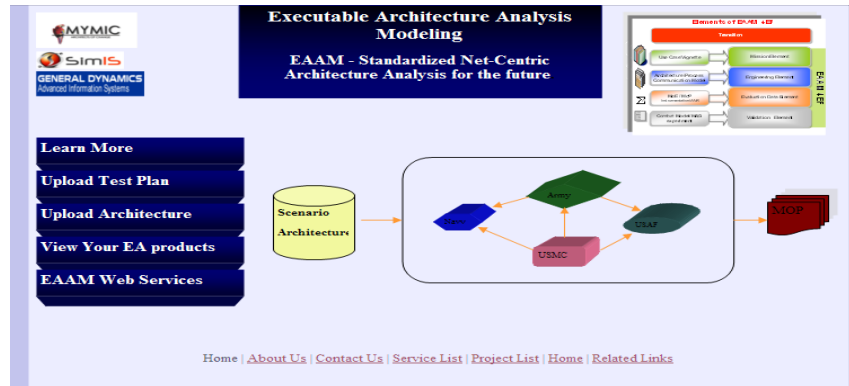
when first logging in and then be selectable through a menu or through a mouse over selection. The user should be able to get more detailed information such as they would get using static DODAF products which have drill down capabilities. In other words the user should get the big picture first and then be able to pick and choose what they need to take a closer look at. A closer look should give more detailed information as shown in Figure 6.



**Figure 5 shows an overall view of the SV-1 systems view**



**Figure 6 shows the users ability to select a portion of the view and get more information with a mouse over**



**Figure 7 shows a possible user interface for EAAM**

The research done to enable the user interface to be scalable, selectable and adaptable is still ongoing at this time. The analysis of data is the key point of the visualization. Pie charts, histograms and bar charts can all be used as well as matrices to help the user make informed decisions about the architecture that they are analyzing. The ability to read, understand, compare and evaluate mission thread architecture products will enhance the design of architectures and allow the optimization of processes. This is just another tool EAAM is using to be able to help the joint community make better choices for the future.

### Summary

Research into developing executable capabilities for network centric testing and evaluation has significant potential to transform the discipline of architecture development into a state where it more closely resembles the approaches which other technical disciplines follow when designing, specifying, and then constructing the artifacts which are the focus of their work. Architectural specifications which support collaboration amongst designers and programmers and with stakeholders who will procure, test and use such systems are needed. However, such architecture specifications and their consequential products need to be dynamic to support these collaborative dialogs and allow non computer scientists to understand how the system is intended to function. Executable architectures hold the potential to achieve that goal and are worthy of further research and development at the investigatory level supported by federal research funding agencies.

### References

AnyLogic 6.2 Advanced <http://www.xjtek.com/anylogic/>

Banks Jerry, Carson John, Nelson Barry, Nicol David, Discrete-Event System Simulation, New Jersey: Pearson, 2005.

Greg Conti, Security Data Visualization: Graphic Techniques for Network Analysis, San Francisco CA.: No Starch Press, 2007

Kuhl, Frederick et.al, eds. Creating Computer Simulation Systems: An Introduction to the High Level Architecture, New Jersey: Prentice Hall Inc., 2000.

Metastorm ProVision <http://www.metastorm.com/products/mpea.asp>

### **Author Biographies**

**Johnny Garcia** is founder and CEO of SimIS, Inc. and a Modeling and Simulation Ph.D. Candidate at Old Dominion University. He has over 19 years of engineering experience that includes systems architecture design, software development, database development, C4I systems development, logistics systems development, and new technology insertion for the Department of Defense. His Dissertation is in the creation of an Executable Architecture Analysis Modeling method discussed in this paper.

### **Acknowledgements**

**Chris Blancett** – is a software engineer at SimIS; Chris received his Bachelor of Science degree from Old Dominion University.