Automation using XML Enhancing DoD Af use for DEVS Based T&E

Saurabh Mittal, José Luis Risco Martín

July 23, 2006
**Basic Concept**

1. Activities
2. Port Info

**DoDAF OV (OV-5, 8, 9)**

**In XML Document Structure format**

- Formulation of Activity-model with defined port-interfaces

**AUTOMATED PROCESS**

**DEVS DTD**

DEVS Atomic DTD
Contains place-holders for encapsulation of Activity-functions/logic

**Merging XML Docs using DEVS-DoDAF-SES**

**DEVS Models in XML**
With Activity Encapsulation (as functions)

**DISTRIBUTED DEVSJAVA EXECUTION**

**Service/Activity SES**

**Dynamic Run-time Coupled model creation**
Using Activity Hierarchy And Decomposition

**SOA middleware**

**Automated XML to JAVA Transformation using DEVS coupled DTD**
DEVS DTDs and their Automated Execution

DEVS Formalism

DEVS Atomic DTD

DEVS Coupled DTD

Atomic Model in XML (underlying JAVAML For behavioral specs)

Coupled Model Scenario in XML (containing couplings And input-injects)

INTEGRATED MODEL DESCRIPTION FILE IN XML

LIBRARY of Atomic/Coupled Models in XML (may be Web-based)

SOA/CORBA/RMI Middleware

Generate JAVA/C++ Code from XML using:
1. XSL
2. XML-to-java

Partition into Coupled/atomic XML descriptions (AUTOMATED)

simulate

DISTRIBUTED or CENTRAL DEVSJAVA/DEVSC++ Execution

Atomic Model in JAVA

JAVA-to-XML Translation using Pre-processing Concepts (Automated)

7/23/2006 2:47:04 PM Mittal and Martin
Implementation

Figure 16: an SOA object capable of DEVS modeling

Figure 17: Automated XML snippet for an Activity Component created with OV-8 (port definitions). Logic is added later or exchanged through SOA implementation.
We now have…

…the Capability to:
1. Transform any DEVS java to XML description and vice- versa
2. Validate any atomic or coupled DEVS model through XML DTDs (i.e. moving towards standardization of code…)
3. Write in pure XML (both atomic and coupled DEVS) and run the simulation on the web OR on machine
4. Collaborate with any XML atomic models ‘remotely’ and create dynamic Coupled models
5. Run the simulation and provide ‘integration’ as a Service in SOA implementation
6. Run in distributed manner using RMI (feature to be included in DTDs)
### DEVS/DoDAF as Basis for Life Cycle Development with integrated Net-based T&E

<table>
<thead>
<tr>
<th>Components &amp; Interfaces</th>
<th>Activity Component Metadata in XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>OV-5,6b,c Capabilities Activities Identification</td>
<td>OV-6a Rule of Engagement</td>
</tr>
<tr>
<td>Operational and Performance Parameters Identification</td>
<td>Rule-based Meta-model Semantic Structure</td>
</tr>
<tr>
<td>Activity Component Identification</td>
<td>Rule Structure Definition and Classification</td>
</tr>
<tr>
<td>Activity Component &amp; Interfaces</td>
<td>OV-6a Rule in Structured English</td>
</tr>
<tr>
<td>Refinement</td>
<td>Automated NLP-based Code-translation</td>
</tr>
</tbody>
</table>

#### Mission Requirements

- AV-1,2 OV1,2 Conceptual Architecture Descriptions
- OV-8, 9 Operational and Performance parameters Identification

#### Model Execution

- Run-time Coupled Model Creation/Deployment using Experimental Frames
- Performance Results & Execution

#### Domain-specific Rules/Policies Definition

- DEVS/DoDAF as Basis for Life Cycle Development with integrated Net-based T&E

### DEVS/DoDAF as Basis for Life Cycle Development with integrated Net-based T&E

- OV1,2 Conceptual Architecture Descriptions
- OV-5,6b,c Capabilities Activities Identification
- Operational and Performance Parameters Identification
- Activity Component Identification
- Refinement

### Activity Component & Interfaces

- Activity Component Identification
- Activity Component & Interfaces

### Components & Interfaces

- OV-5,6b,c Capabilities Activities Identification
- Operational and Performance Parameters Identification
- Activity Component Identification
- Refinement

### OV-6a Rule of Engagement

- Rule-based Meta-model Semantic Structure
- Rule Structure Definition and Classification
- OV-6a Rule in Structured English

### Automated NLP-based Code-translation

- DEVS/DoDAF as Basis for Life Cycle Development with integrated Net-based T&E

### Distributed Net-centric Platform

- DEVS Activity Component/Model
- DEVS model Semantics/Time-advance criticalities
- XML-DEVS Dynamic translation

### DEVS Activity Component/Model Specified in XML

- XML-DEVS Dynamic translation
- DEVS model Semantics/Time-advance criticalities

### Web Model Repository (XML based)

- XML-DEVS Dynamic translation
- DEVS model Semantics/Time-advance criticalities

### Run-time Coupled Model Creation/Deployment using Experimental Frames

- Performance Results & Execution
- Distributed Net-centric Platform

### 7/23/2006 2:47:04 PM

### Ready for Systems Deployment
DoDADF and T&E Strategy

NR-KPP
Testing & Evaluation
(T&E) Strategy

OV-5,6
Capabilities
Activities
Identification

Step 1

VALIDATE the exchanges in
Operational Test (OT)
Environment

Step 2

OV-8, 9
Operational and
Performance
parameters
Identification

VERIFY Data-exchange
from information gathered
e.g. SV-6

Key Interface Profile
- Refined OVs and SVs
- Interface Control
document/Specs
- Engg. Mgmt. Plan
- Config. Mgmt. Plan
- TV with SV-TV bridge
- Procedures for
Standards Conformance & Testing

SV-6,7
Systems Data
Exchange matrix &
Performance
parameters
Application Snapshot

Files in the Current example
- Java Atomic Models
  - DEVSM coupled model
- Contents of Source dir
- Contents of Target dir (the generated files)
- Integrate a DEVSM coupled scenario with multiple Java atomic files in source folder
- Contents of the Generated file in the Target folder as a result of Operation on the source file

Select the file to View and Operate
- Convert a JAVA atomic model to DEVSM description etc.

View the Source And Target folder files in DEVSM and Java format
- Contents of Target dir
- View Src/Target

Example for DEVSM Implementation
- Source Dir
- Target Dir
- Contents of Source dir

Convert a JAVA atomic model to DEVSM description etc.

Integrate Coupled XML with Atomic Java (in Source dir)
- Atomic Java
  - Scope.java
  - Function.java
  - Integrator.java
- Target Dir
  - AttractorLorenz.xml
  - Scope.java
  - Function.java

Contents of the Generated file in the Target folder as a result of Operation on the source file

Note
- Buttons above are enabled based on applicable operations
- View the Source And Target folder files in DEVSM and Java format
- Contents of Target dir
- View Src/Target

Integrate a DEVSM coupled scenario with multiple Java atomic files in source folder
- Contents of the Generated file in the Target folder as a result of Operation on the source file
- Contents of Target dir
- View Src/Target

Convert a JAVA atomic model to DEVSM description etc.

Integrate Coupled XML with Atomic Java (in Source dir)
- Atomic Java
  - Scope.java
  - Function.java
  - Integrator.java
- Target Dir
  - AttractorLorenz.xml
  - Scope.java
  - Function.java

Contents of the Generated file in the Target folder as a result of Operation on the source file

Note
- Buttons above are enabled based on applicable operations
- View the Source And Target folder files in DEVSM and Java format
- Contents of Target dir
- View Src/Target
References

Working Papers
• Jose Luis Risco Martin, Saurabh Mittal, et al., A W3C XML Schema for DEVS Scenarios
• Saurabh Mittal, Jose Luis Risco Martin, DEVSML: Automating M&S with JAVAML
• Saurabh Mittal, Jose Luis Risco Martin, Bernard P. Zeigler, Automating DEVS-DODAF Test & Evaluation Methodology using DEVSML

Referenced Papers
• Vladimir Janousek, Petr Polasek, Pavel Slavicek, Towards DEVS Meta Language

Download .exe available at: www.u.arizona.edu/~saurabh/devsml/devsml.html