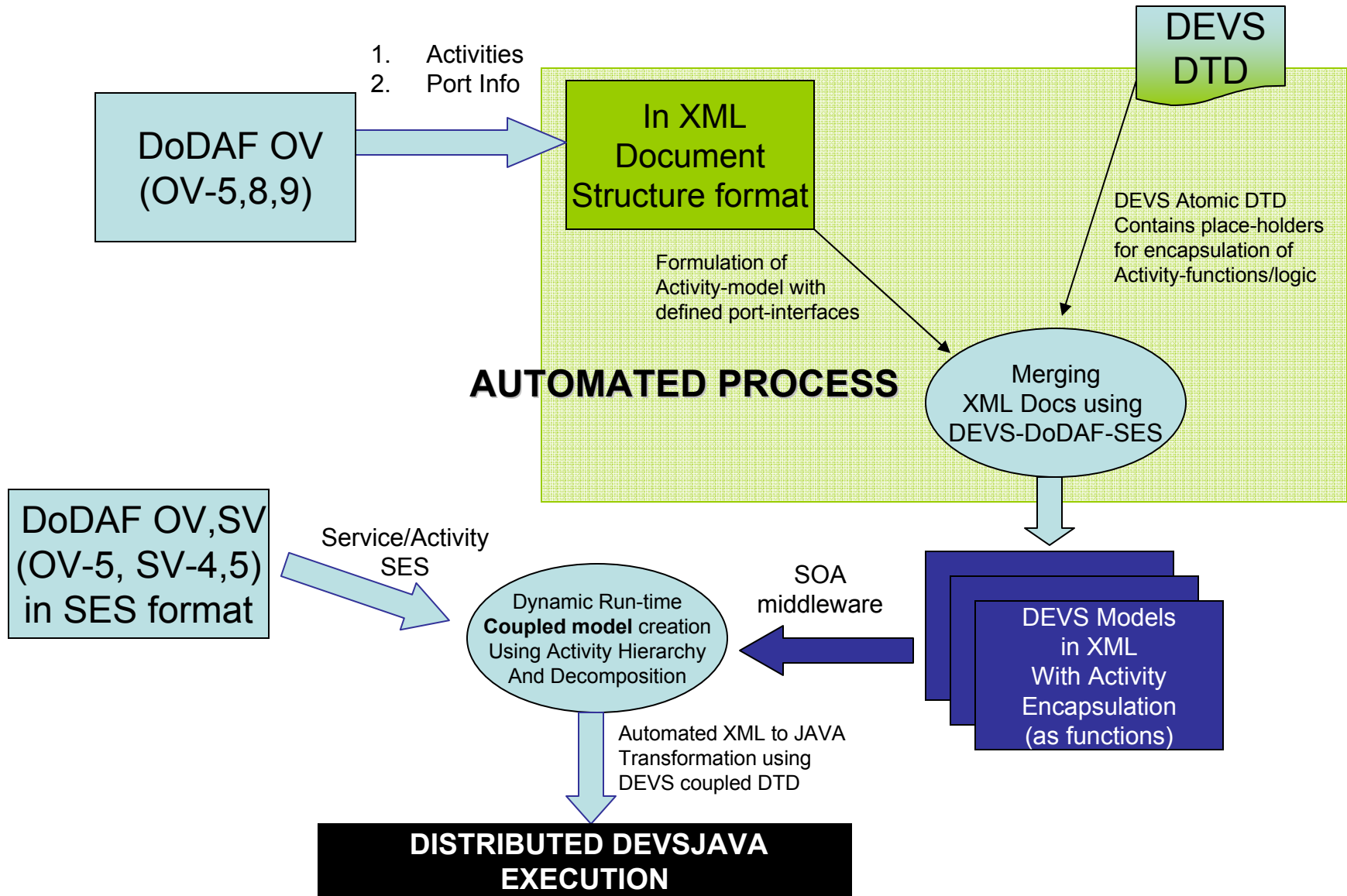


Automation using XML Enhancing DoDAF use for DEVS Based T&E

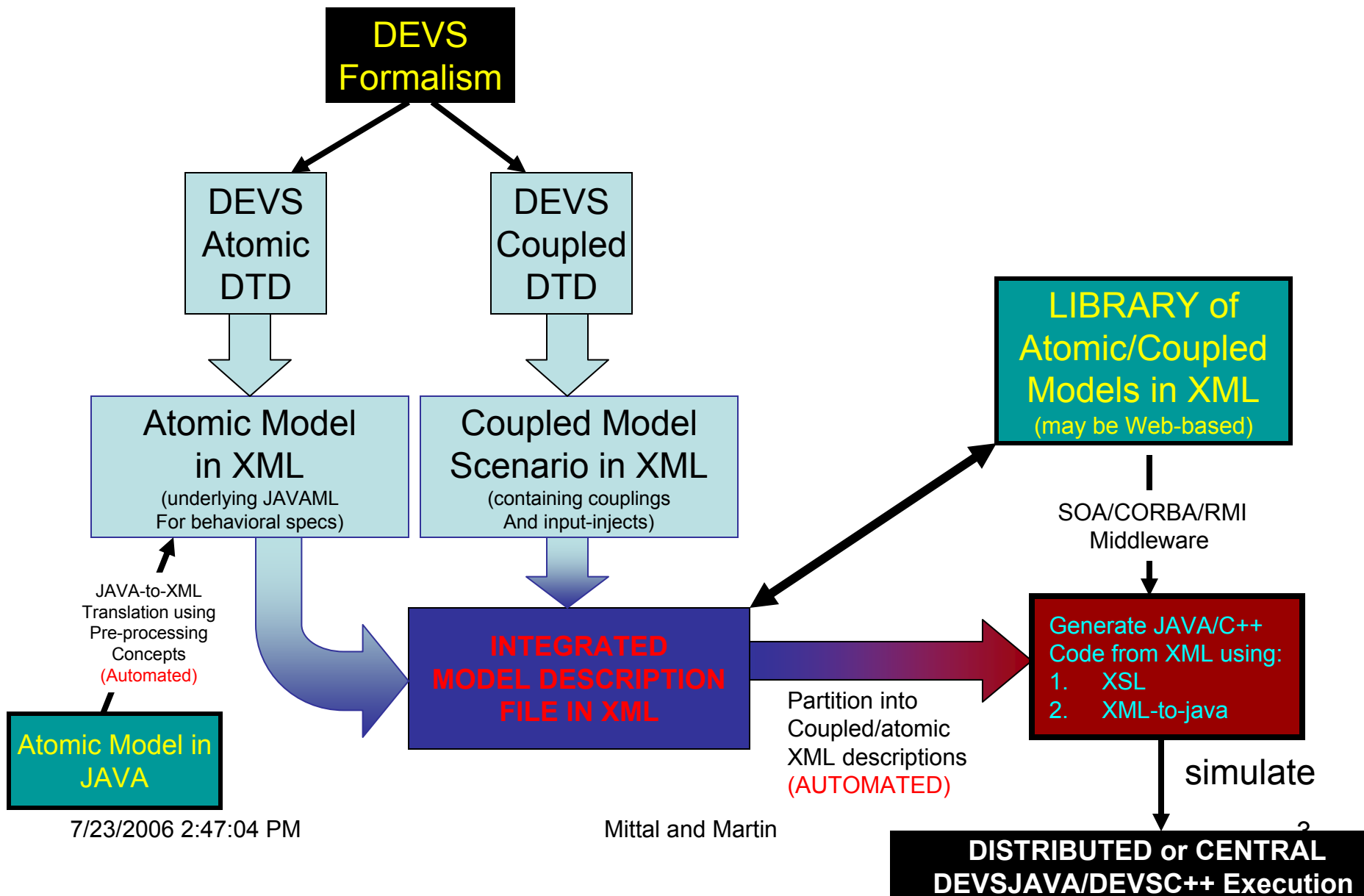
Saurabh Mittal, José Luis Risco Martín

July 23, 2006

Basic Concept



DEVS DTDs and their Automated Execution



Implementation

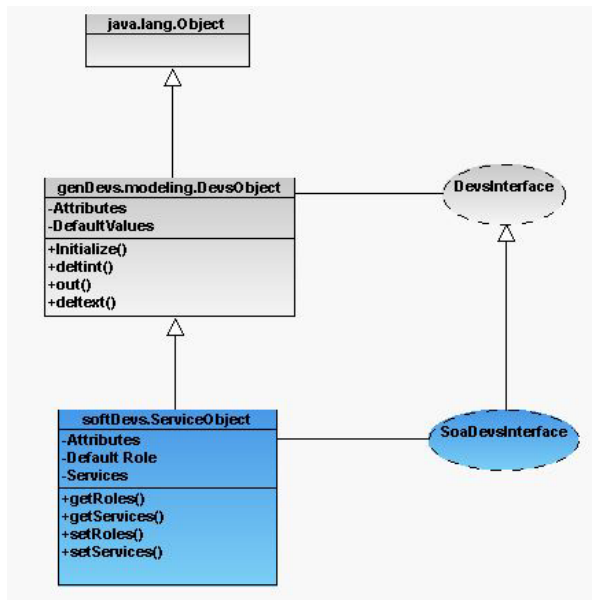


Figure 16: an SOA object capable of DEVS modeling

```

<?xml version="1.0" encoding="UTF-8"?>
  <xml-body>
    <model>
      <atomic>
        <name>Hello</name>
        <params> </params>

        <construct>
          <args> </args>
          <ports>
            <inports>
              <inport>in</inport>
            </inports>
            <outports>
              <outport>out</outport>
            </outports>
          </ports>
        </construct>

        <initialize>
        </initialize>

        ...

      </atomic>
    </model>
  </xml-body>

  <services>
    <function>
      <access> public </access>
      <return> int </return>
      <inport> in </inport>
      <outport> out </outport>
      <fname> decrement() </fname>
    </function>
  </services>
  
```

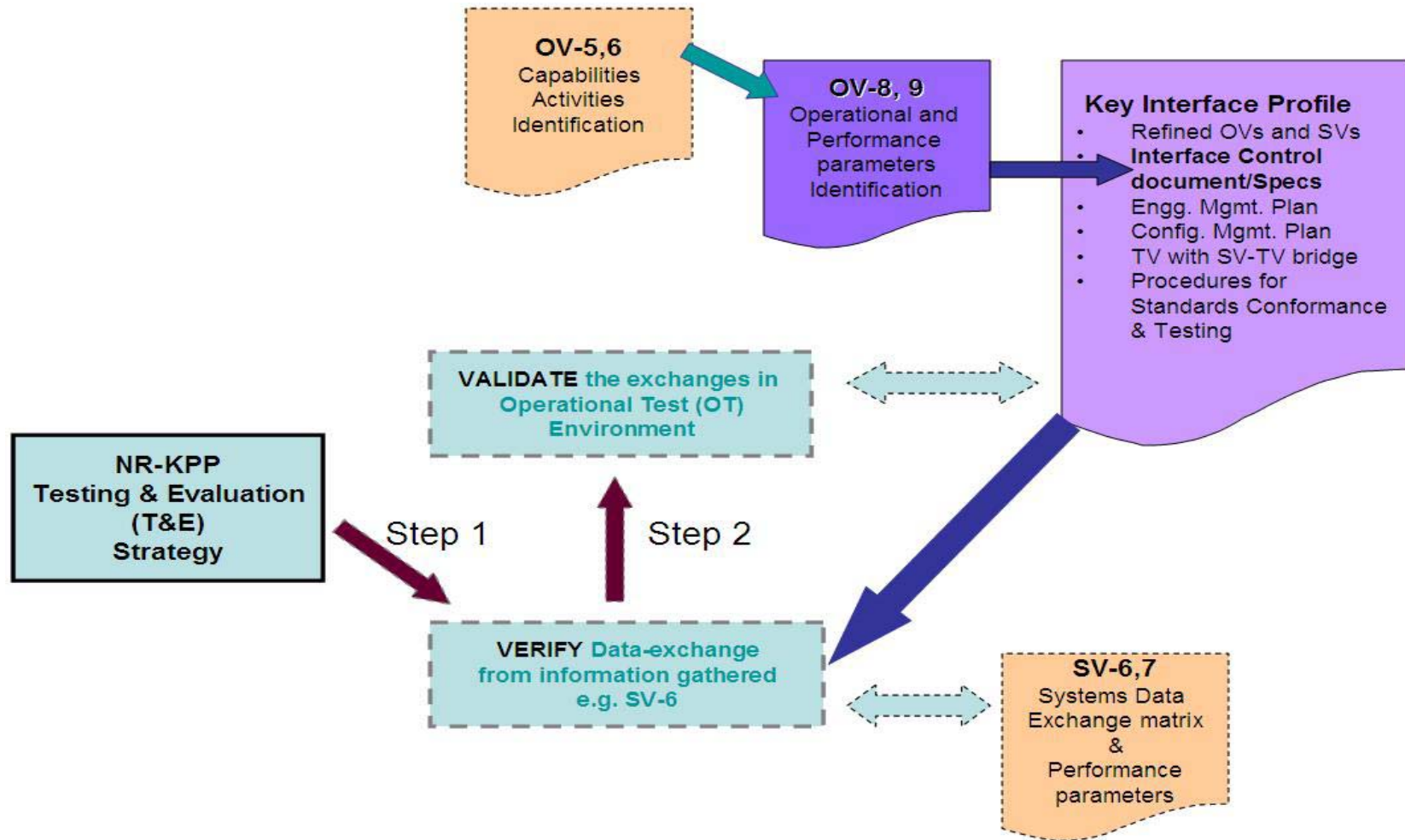
Figure 17: Automated XML snippet for an Activity Component created with OV-8 (port definitions). Logic is added later or exchanged through SOA implementation.

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)

DoDAF and T&E Strategy



Application Snapshot

The screenshot shows the DEVML Model Example application window. The title bar reads "DEVML Model Example". The main content area is titled "Example for DEVSML Implementation".

Files in the Current example: A list of files is shown, including "src\Scope.java", "src\Function.java", "src\Integrator.java", "src\AtractorLorentz.xml", "target\AtractorLorentz.xml", "target\Function.xml", and "target\Integrator.xml".

Source Dir: A list of files in the source directory: "Function.java", "Integrator.java", "Scope.java", and "AtractorLorentz.xml".

Target Dir: A list of files in the target directory: "AtractorLorentz.xml", "Function.xml", "Integrator.xml", and "Scope.xml".

Integrate Coupled XML with Atomic Java (in Source dir): A section with a "Show XML" button and a text area containing XML code for a DEVSML scenario.

Convert a JAVA atomic model to DEVSML description etc.: A yellow callout pointing to the "Atomic" and "Coupled" radio buttons. The "Atomic" button is selected.

Integrate a DEVSML coupled scenario with multiple Java atomic files in source folder: A yellow callout pointing to the "Integrate Coupled XML with Atomic Java" section.

Contents of the Generated file in the Target folder as a result of Operation on the source file: A callout pointing to the XML code in the "Integrate Coupled XML" section.

Contents of Source dir: A callout pointing to the "Source Dir" list.

Contents of Target dir (the generated files): A callout pointing to the "Target Dir" list.

Select the file to View and Operate: A callout pointing to the file list in the "View Src/Target" section.

View the Source And Target folder files in DEVSML and Java format: A callout pointing to the "View Src/Target" section.

Note: Buttons above are enabled based on applicable operations.

XML Code:

```
<devs>
<scenario>
<coupled model="AtractorLorentz" name="root">
<atomic model="Function" name="fx">
<inputs>
<port name="x" type="java.util.ArrayList"/>
<port name="u" type="java.util.ArrayList"/>
</inputs>
<states>
<state name="_f" type="java.util.ArrayList">
<initialization>
<list>
<parameter type="java.lang.String" value="10*(x_1-x_0)"/>
<parameter type="java.lang.String" value="x_0*(28-x_2)>
<parameter type="java.lang.String" value="x_0*x_1-(8/3)*
</list>
</initialization>
</state>
<state name="_u" type="java.util.ArrayList"/>
<state name="_x" type="java.util.ArrayList"/>
<state name="_exp" type="java.lang.String"/>
</states>
<outputs>
<port name="out" type="java.util.ArrayList"/>
</outputs>
</atomic>
```


References

Working Papers

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

Referenced Papers

- Vladimir Janousek, Petr Polasek, Pavel Slavicek, *Towards DEVS Meta Language*
- **Saurabh Mittal**, Amit Mitra, Amar Gupta, Bernard P. Zeigler, *Strengthening OV-6a Semantics with Rule-based Meta-models in DEVS/DoDAF Based Life-cycle Architectures Development*, IEEE-Information Reuse and Integration ([IRI06](#)) Conference, [special section on DoDAF](#), Hawaii, September 2006 ([pdf](#))
- **Saurabh Mittal**, *Extending DoDAF to Allow Integrated DEVS-based Modeling and Simulation*, (under review) submitted to Journal of Defense Modeling and Simulation [JDMS](#), 2005 ([pdf](#))
- **Saurabh Mittal**, Eddie Mak, James J. Nutaro, *DEVS-Based Dynamic Model Reconfiguration and Simulation Control in the Enhanced DoDAF Design Process*, (under review) submitted to Journal of Defense Modeling and Simulation [JDMS](#), 2005 ([pdf](#))
- Bernard P. Zeigler, **Saurabh Mittal**, *Enhancing DoDAF with a DEVS-based System Lifecycle Development Process*, In Proceedings of IEEE International Conference on Systems, Man and Cybernetics, [SMC05](#), Hawaii 2005 ([pdf](#))

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