Model-Driven Systems Engineering for Netcentric System of Systems With DEVS Unified Process

Saurabh Mittal, PhD
Dunip Technologies, L3 Communications, USA

Jose L. Risco Martin, PhD
Universidad Complutense de Madrid, Spain

Winter Simulation Conference, Dec. 8-11, 2013, Washington, DC
Outline

• Model-based and Model-driven: Flavors
• Metamodeling and Domain-Specific Languages
• Theory of M&S and Levels of Systems specifications
• DEVS Unified Process and DEVSML Stack
• Netcentric SoS: EDA and DEVS together
Model-Based and Model-Driven Flavors

• MBE/MBD: Model-Based Engineering/Design
  • 1980s: Wymore and Zeigler
  • Design, development, integration, validation, verification, testing, documentation, maintenance

• MBSE: Model-Based Systems Engineering
  • Analysis and Design phases, systems complexity, team communication

• MDE: Model-Driven Engineering
  • 2000s
  • Focus on Transformations and metamodels: Usage of models in various phases

• MDA: Model-Driven Architecture
  • 2000s, OMG
  • MOF: Guidelines for specifying and structuring models: context independence

• MDD/MDSD: Model-Driven Software Development
  • 1990s: OMG, Eclipse, Microsoft and others

• MIC: Model Integrated Computing
  • 1990s: ISIS
  • Open integration framework to support formal analysis tools, verification techniques and model transformations
Models in Systems/Software Engineering

MDSE: Model-Driven Systems Engineering
Use of MDE to enhance the capabilities inherent in MBSE
MDE
Key Enabler promoting automated transformations

• Metamodeling
  M1, M2, and M3 Levels

• Domain Specific Languages
  • Defined at M2 Level
  • Oriented to a problem domain/context
  • Metamodelling process is called Domain Specific Modeling (DSM)
Theory of Systems M&S: Concepts

- System Specification Formalisms: Continuous or Discrete
  - DESS, DTSS, Quantized
- Hierarchy of Systems Specifications
  - Closed under composition

<table>
<thead>
<tr>
<th>Level</th>
<th>Name</th>
<th>System Specification at this level</th>
<th>Elements from the Framework for M&amp;S</th>
<th>Verification and Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Coupled Systems</td>
<td>Systems built from component systems with a coupling recipe</td>
<td>Model, Simulator, Experimental Frame</td>
<td>Structural Validity, simulator correctness</td>
</tr>
<tr>
<td>3</td>
<td>I/O System Structure</td>
<td>System with state and transitions to generate the behavior</td>
<td>Model, Simulator, Experimental Frame</td>
<td>Predictive Validity</td>
</tr>
<tr>
<td>2</td>
<td>I/O function</td>
<td>Collection of input/output pairs partitioned according to initial state</td>
<td>Model, Source System</td>
<td>Replicative Validity</td>
</tr>
<tr>
<td>1</td>
<td>I/O behavior</td>
<td>Collection of input/output pairs from external black-box view</td>
<td>Model, Source System</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>I/O frame</td>
<td>Input and output variables and ports together with values over a time base</td>
<td>Source System</td>
<td></td>
</tr>
</tbody>
</table>

12/9/2013 Mittal and Martin, WSC’13, Washington, DC
Theory of Systems M&S: Concepts (2/2)

• Source-System, Model, Simulator, Experimental-Frame
Object or Model?

- Separation of Model and Simulator: a critical requirement
- Model develops abstractions and simulator executes a model
- The Abstraction chain, layered, hierarchy
- Model transformations
- Semantic anchoring
- Structure and Behavior
DEVSML Stack: Netcentric DEVS Virtual Machine

DEVS Modeling Language (DEVS PIM DSL) (Atomic and Coupled models)

DEVS Middleware (Standards compliant API)

DEVS / SOA

Net-centric Infrastructure (SOA) / Cloud

DEVS / JAVA 192.168.1.100
DEVS / C++ 192.168.1.101
DEVS / .NET 192.168.1.101
Non-DEVS eg. MATLAB

M2DEVS transformation
M2M transformation
M2DEVSML transformation

End User client
Server-side architecture

Mittal and Martin, WSC'13, Washington, DC
DEVS Unified Process

[Diagram showing the DEVS Unified Process with various categories and connections such as DSLs, M2DEVSMML, M2DEVSM, Translations, DEVS Behavior Requirements at Lower Levels of System Specifications, DEVS Structural Requirements at Higher Levels of System Specifications, Dynamical Systems Theory, Simulation Execution DEVS/SoA, Real-Time Execution, Models to Web Services/DEVs Agents, Transparent Simulators, Verification & Validation, DEVS Experimental Frames, Platform Specific Models, Platform Independent Models, Test Models/Observers/Federations, DoDAF-Based Scenario, BPMN-BPEL-Based Scenario, Message-Based Restricted NLP, UML-Based, State-Machines (FDDEVS).]
Spiral nature of DUNIP

Increasing:
1. Complexity
2. Integration
3. Requirements
4. Components
5. Tests
6. Validity
# MBSE/DUNIP with other MB/MD Flavors

<table>
<thead>
<tr>
<th>Features</th>
<th>System/Software Engineering approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MBE</td>
</tr>
<tr>
<td>Use of DSLs</td>
<td>Y</td>
</tr>
<tr>
<td>Alignment with Systems theory</td>
<td>Y</td>
</tr>
<tr>
<td>DSL representation with metamodeling</td>
<td>-</td>
</tr>
<tr>
<td>Guidance for model transformations</td>
<td>-</td>
</tr>
<tr>
<td>Support for component reusability</td>
<td>Y</td>
</tr>
<tr>
<td>Code generation/execution</td>
<td>Y</td>
</tr>
<tr>
<td>Code deployment mechanisms</td>
<td>Y</td>
</tr>
<tr>
<td>Tool support for overall process</td>
<td>-</td>
</tr>
</tbody>
</table>

12/9/2013 Mittal and Martin, WSC’13, Washington, DC
Netcentric Event Driven Architectures

• SOA:
  • The Key Enabler as events are structured rather than just messages in discrete event systems

• Granular at Event level: Have semantics associated

• Functional components
  • Producer
  • Consumer
  • Processor
  • Reaction (automate/human)
  • Processing Backbone (ESB/Cloud)
Event Processing in EDA

• SEP: Simple Event Processing
  • Exclusively processed and may not have event reactions

• ESP: Event-Stream Processing
  • Events have Temporal nature and multiple correlated events may elicit a reaction

• CEP: Complex Event Processing
  • Multiple ESP on different time scales with meaningful logical reactions
  • Pattern matching on information sets
Contrasting EDA and DEVS Systems Hierarchy

- EDA is a software paradigm and results in real-time event-driven “system” as a whole
- No framework to manage abstract time i.e. there is no simulator
- EDA is stateless: State travels with event

<table>
<thead>
<tr>
<th>Level</th>
<th>Name</th>
<th>EDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Coupled systems</td>
<td>Does not exist. There is no containment to specify system hierarchy.</td>
</tr>
<tr>
<td>3</td>
<td>I/O System</td>
<td>Does not exist</td>
</tr>
<tr>
<td>2</td>
<td>I/O Function</td>
<td>Complex event processing</td>
</tr>
<tr>
<td>1</td>
<td>I/O Behavior</td>
<td>Event stream processing</td>
</tr>
<tr>
<td>0</td>
<td>I/O Frame</td>
<td>Simple Event processing</td>
</tr>
</tbody>
</table>
EDA and DEVS Together
Conclusions

• MBE and MBSE has been in use since 80s

• Object-oriented software engineering led to the emergence of MDE and various other paradigms such as MDD, MDSD and MIC

• DEVS Formalism pioneered MBE/MBSE and largely used for complex dynamical systems engineering

• DUNIP: technological advancement of DEVS incorporates MDE with DEVS resulting in MDSE

• Advanced tooling led to DEVSML stack incorporating MDE concepts
Conclusions (2/2)

• DSLs such as UML, SySML, DoDAF, BPMN through the DEVSML stack become executable through M2M, M2DEVS and M2DEVSML transformations

• DEVSML resulted in a netcentric DEVS Virtual Machine for fast deployment and transparent simulation framework

• Standards in netcentric environment led to paradigms such as SOA and EDA.

• EDA and DUNIP together brings M&S to complex netcentric environments

• Acknowledges the role of end-user as a DSL designer to its role as a event reactionary thereby transforming a netcentric SoS to a Complex Adaptive System (CAS)
Questions/Comments