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

Universedad Complutense de Madrid, Spain

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

Outline

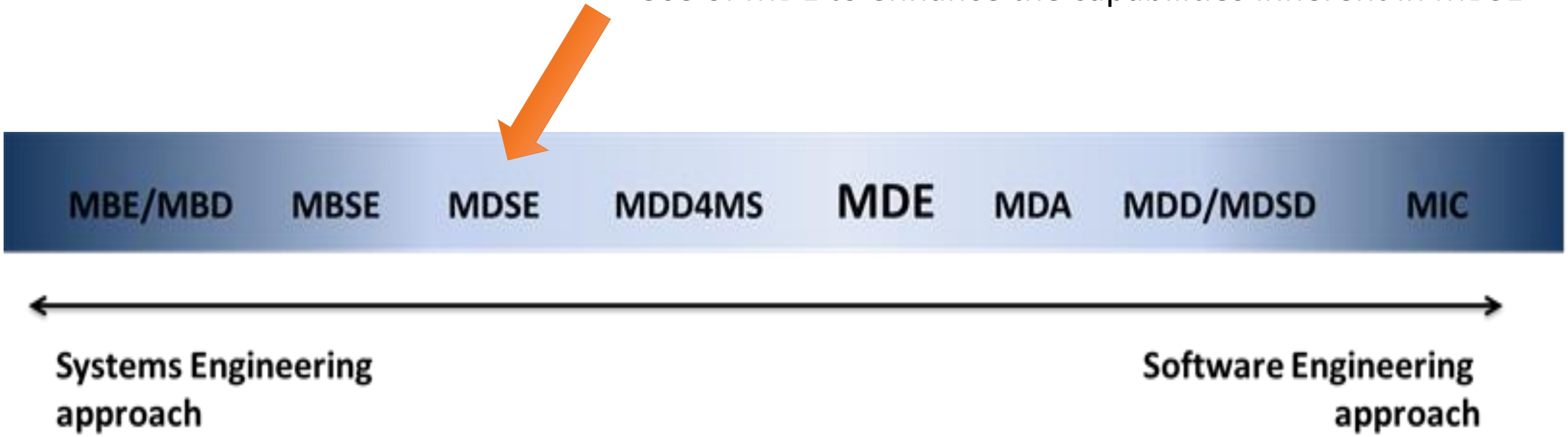
- Model-based and Model-driven: Flavors
- Metamodeling and Domain-Specific Languages
- Theory of M&S and Levels of Systems specifications
- DEVS Unified Process and DEVSML Stack
- Netcentric SoS: EDA and DEVS together

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

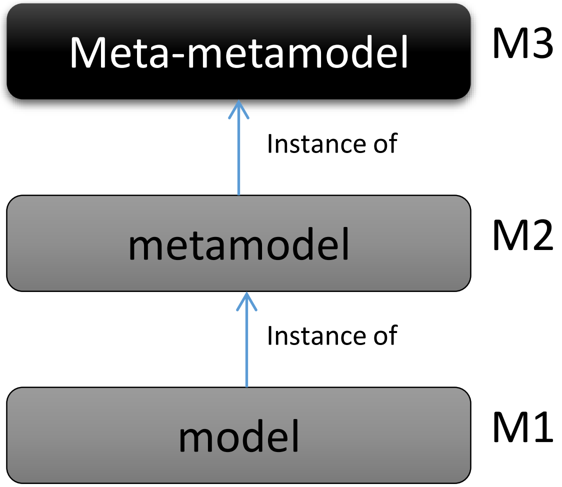


MDE
Key Enabler promoting automated transformations

- Metamodeling

  M1, M2, and M3 Levels

- Domain Specific Languages
  - Defined at M2 Level
  - Oriented to a problem domain/context
  - Metamodeling process is called Domain Specific Modeling (DSM)



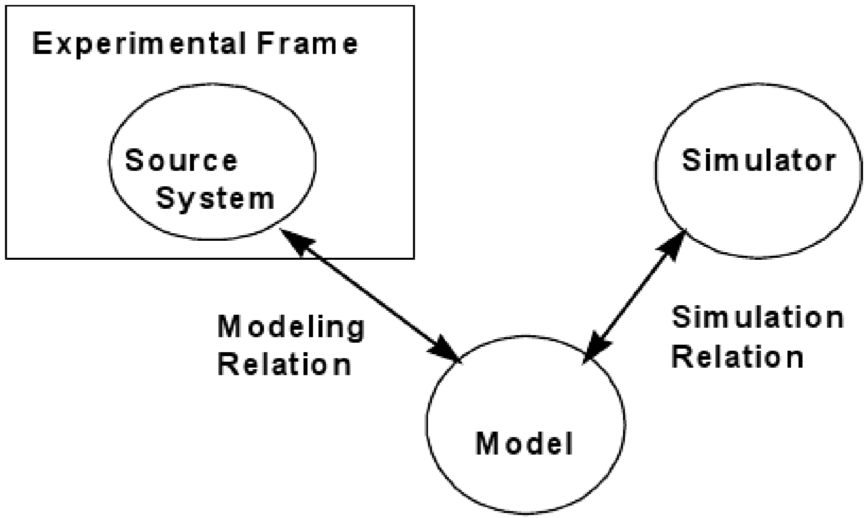
Theory of Systems M&S: Concepts (1/2)

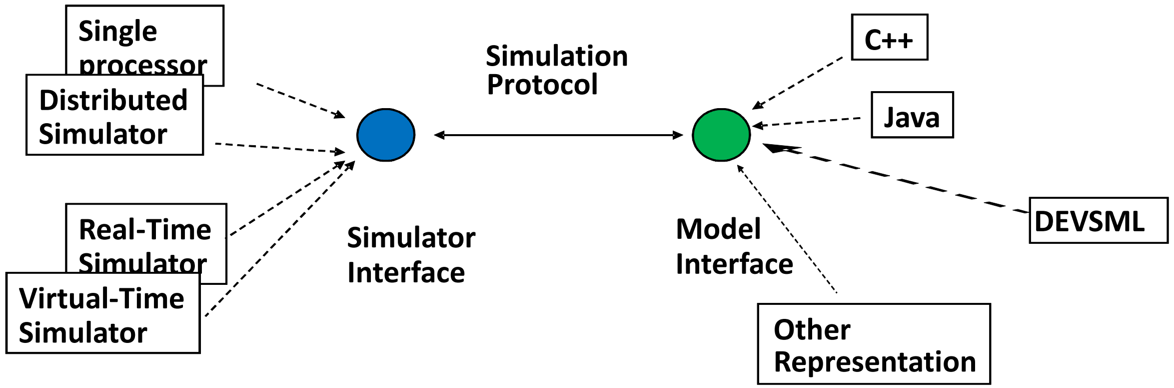
- System Specification Formalisms: Continuous or Discrete
  - DESS, DTSS, Quantized
- Hierarchy of Systems Specifications
  - Closed under composition

<table>
<tr><th>Level</th><th>Name</th><th>System Specification at this level</th><th>Elements from the Framework for M&S</th><th>Verification and Validation</th></tr>
<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 rowspan="2">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></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>Predictive 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>Replicative Validity</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>
</table>

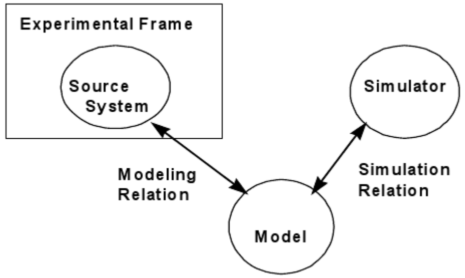
Theory of Systems M&S: Concepts (2/2)

- Source-System, Model, Simulator, Experimental-Frame

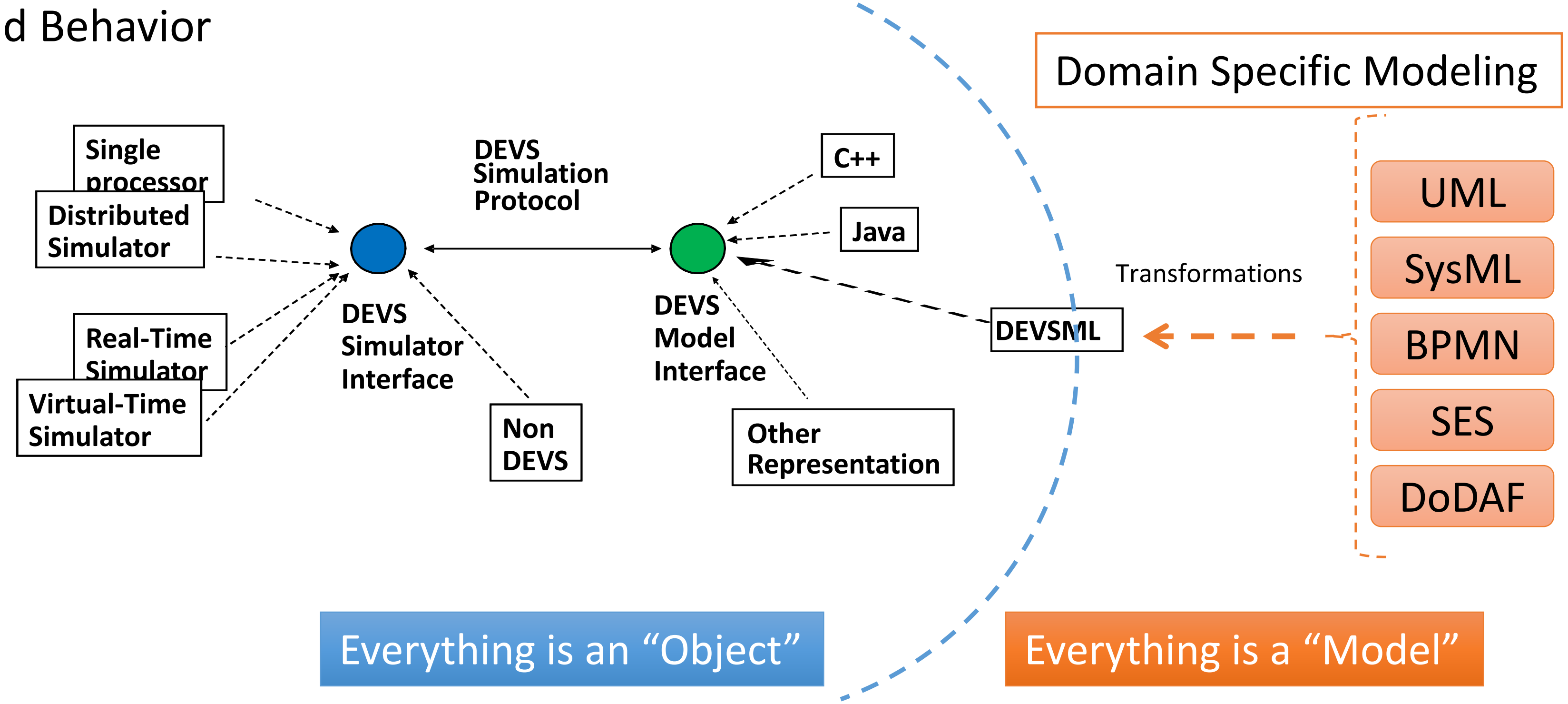




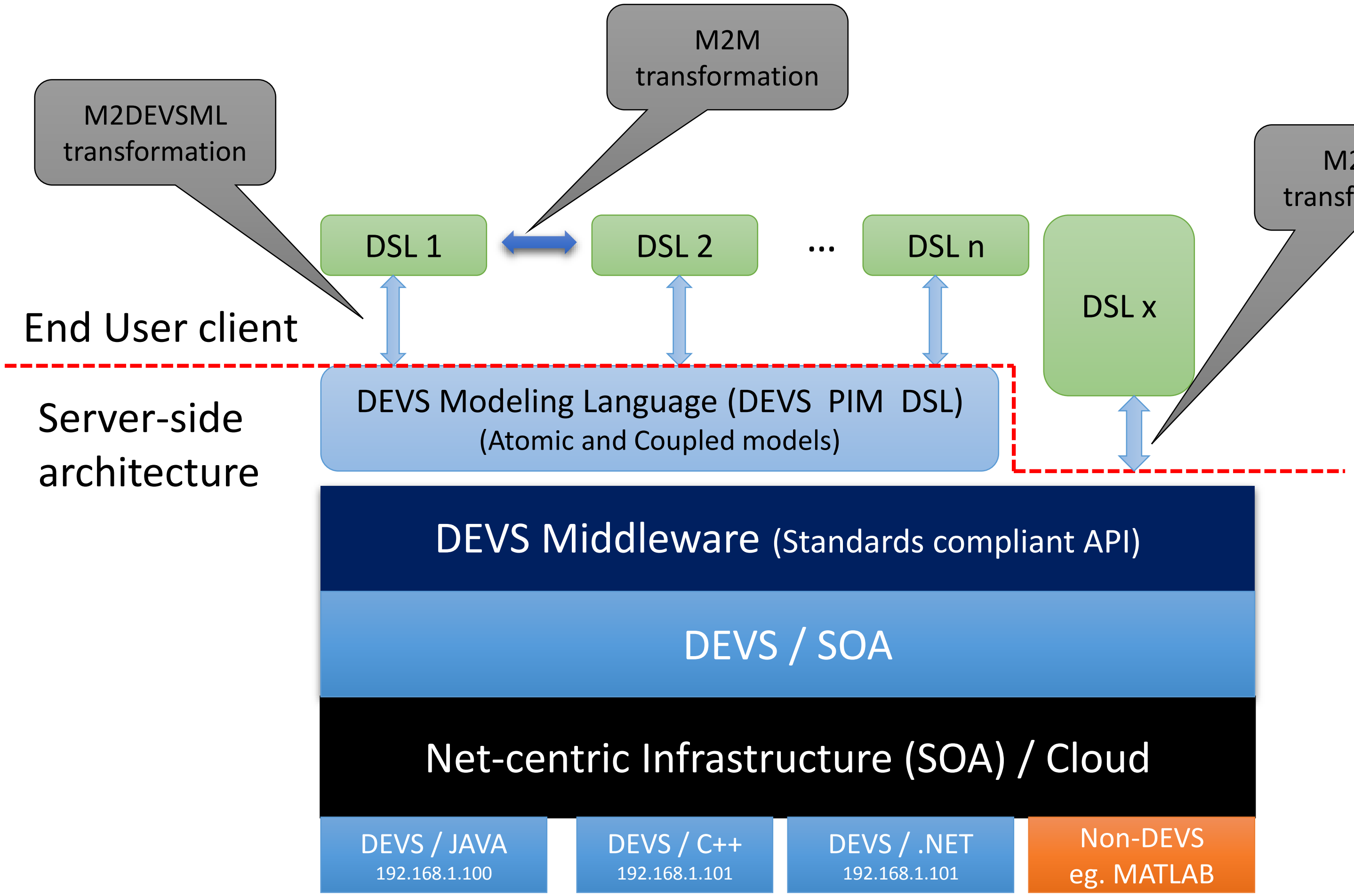
Object or Model?



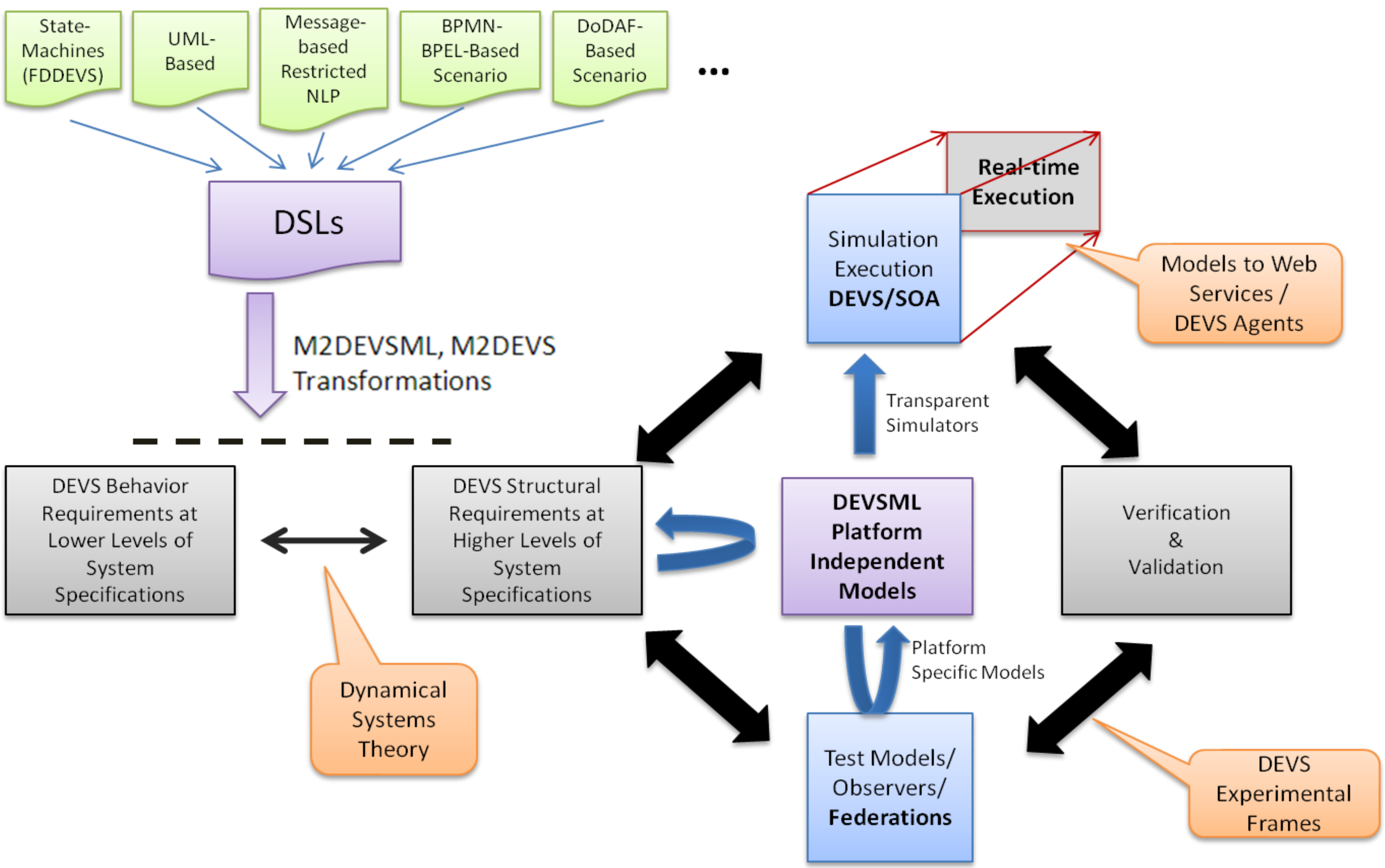
- Separation of Model and Simulator: a critical requirement
- Model develops abstractions and simulator executes a model
- The Abstraction chain, layered, hierarchy
- Model transformations
- Semantic anchoring
- Structure and Behavior



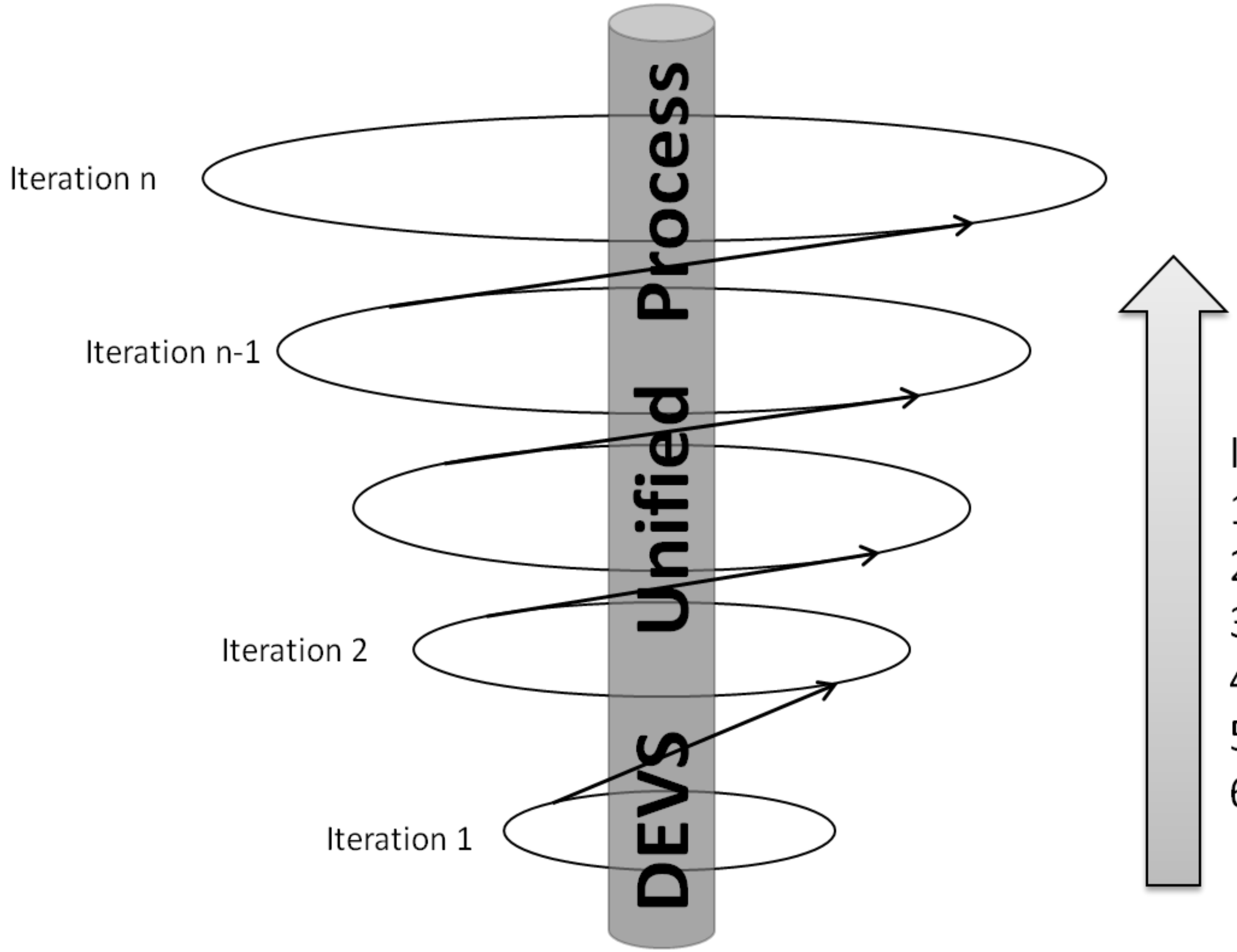
DEVSML Stack: Netcentric DEVS Virtual Machine



DEVS Unified Process



Spiral nature of DUNIP



Increasing:
1. Complexity
2. Integration
3. Requirements
4. Components
5. Tests
6. Validity

MBSE/DUNIP with other MB/MD Flavors

Features	System/Software Engineering approaches							
	MBE	MBSE	MDSE / DUNIP	MDD4MS	MDE	MDD / MDSD	MDA	MIC
Use of DSLs	Y	Y	Y	Y	Y	Y	Y	Y
Alignment with Systems theory	Y	Y	Y	-	-	-	-	-
DSL representation with metamodeling	-	-	Y	Y	Y	Y	Y	Y
Guidance for model transformations	-	-	Y	Y	Y	Y	Y	Y
Support for component reusability	Y	Y	Y	Y	-	-	-	Y
Code generation/execution	Y	Y	Y	Y	-	Y	Y	Y
Code deployment mechanisms	Y	Y	Y	-	-	Y	-	Y
Tool support for overall process	-	-	Y	Y	Y	Y	-	Y
Applicable to all domains	Y	Sys. Engg.	Sys. Engg.	Y	Y	Soft. Engg.	Soft. Engg.	Soft. Engg.

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

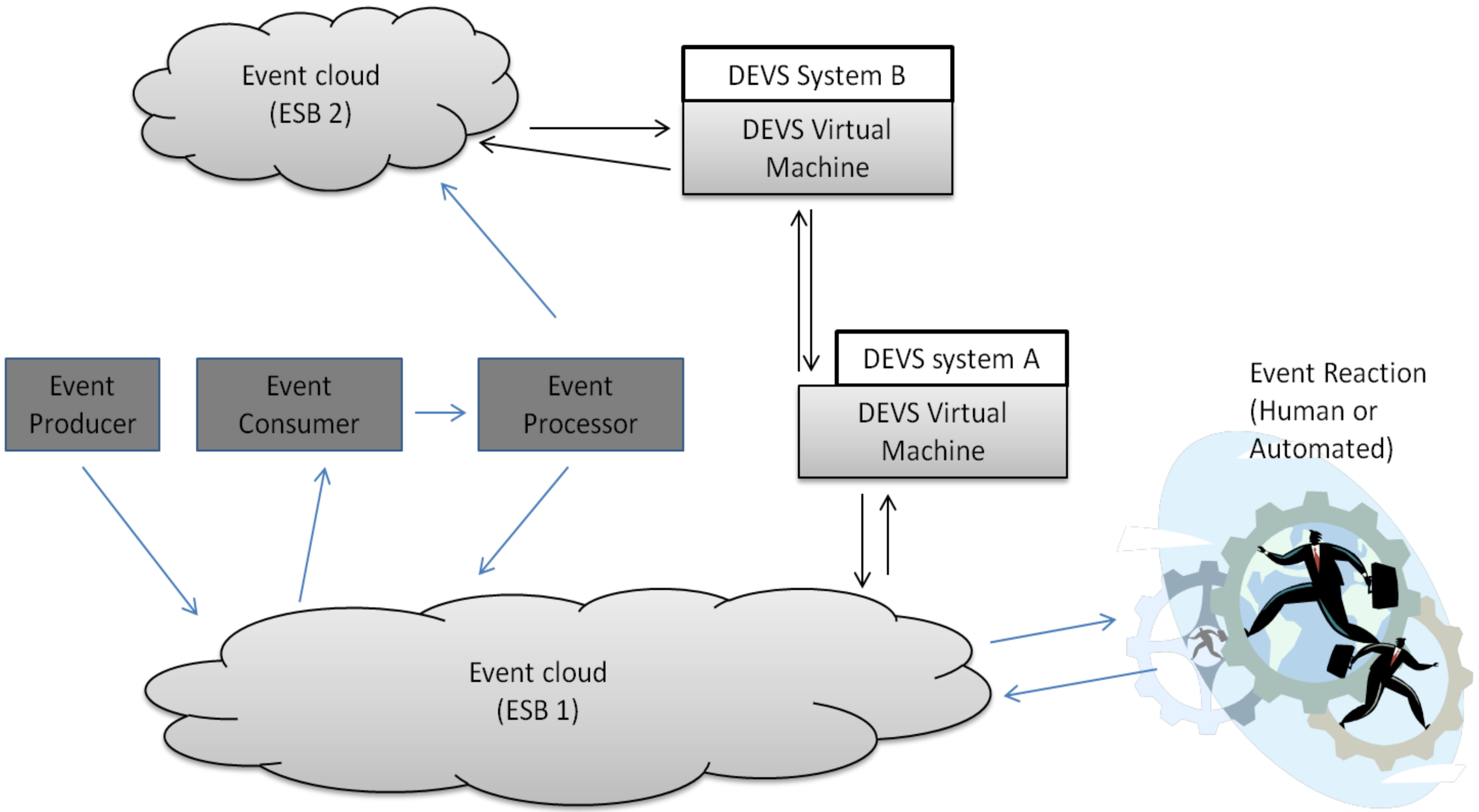
- SEP: Simple Event Processing
  - Exclusively processed and may not have event reactions
- ESP: Event-Stream Processing
  - Events have Temporal nature and multiple correlated events may elicit a reaction
- CEP: Complex Event Processing
  - Multiple ESP on different time scales with meaningful logical 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

Level	Name	EDA
4	Coupled systems	Does not exist. There is no containment to specify system hierarchy.
3	I/O System	Does not exist
2	I/O Function	Complex event processing
1	I/O Behavior	Event stream processing
0	I/O Frame	Simple Event processing

EDA and DEVS Together



Conclusions (1/2)

- 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 enviroments
- 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