

Systems and Software engineering with/out Simulation: State of the Art and Way Forward

Sept 25, 2019

Bernard P. Zeigler
Prof. Emeritus U. Arizona
Chief Scientist, RTSync Corp.
IEEE 2019-2021 Distinguished Speaker

Outline

- Modeling and Simulation (M&S) represents a core capability needed to address today's complex, adaptive, systems of systems (SoS) engineering challenges.
- The limitations of Model-Based Systems Engineering (MBSE) include limited capability to develop multifaceted models, as well as their analysis with computationally powerful and correct simulation engines.
- Software engineering has become a primary implementation for SoS development so it must also be brought into the discussion.
- We discuss potential for closer integration between the three streams.

Empowering the people who drive technology.

Bernard P. Zeigler



IEEE COMPUTER SOCIETY

IEEE

Who We Are

- The IEEE Computer is the world's home for computer science, engineering, and technology. Known as the organization that empowers the people who drive technology, the IEEE Computer Society brings forward-thinking technology professionals together to discover the next technological innovation, develop international standards, form communities, and share knowledge.



Our Vision:

To be the leading provider of technical information, community services, and personalized services to the world's computing professionals.

IEEE Computer Society Today

Global Reach



60,000+ Members



480 Chapters



168 Countries

Technical Breadth

200 Conferences

600,000+ Trusted Technical Documents

30+ Top-cited Periodicals

121+ Active Standards

Student Scholarships

$40,000+ Available Each Year

Richard E. Merwin Student Ambassador

- 36, $1000 Scholarships Awarded
- Deadlines: April 30 & September 30
- Designed to recognize and reward active student volunteer leaders in student branches or student branch chapters

Lance Stafford Larson Paper Contest

- 2, $500 Scholarships Awarded
- Deadline October 31

Upsilon Pi Epsilon Honor Society

- 4, $1,000 Scholarships Awarded
- Deadline: October 31

Student Scholarships

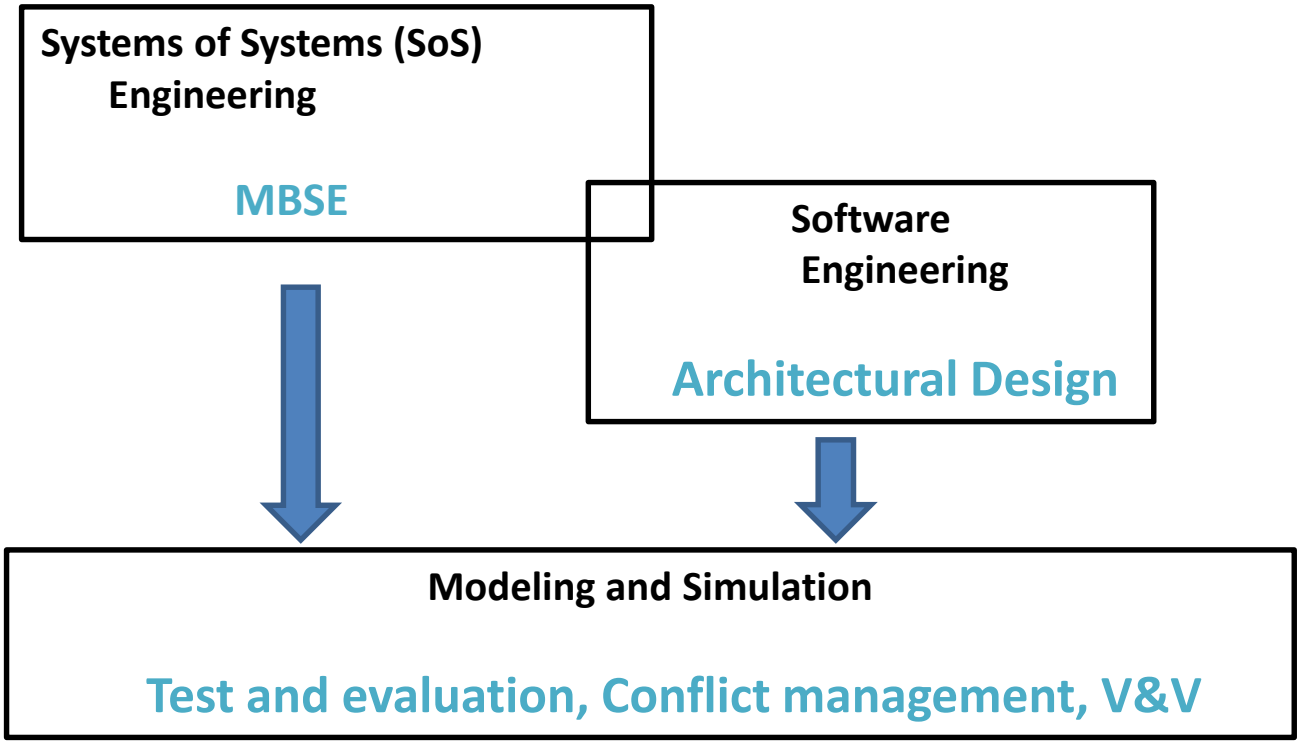
$40,000+ Available Each Year

Distinguished Visitors Program

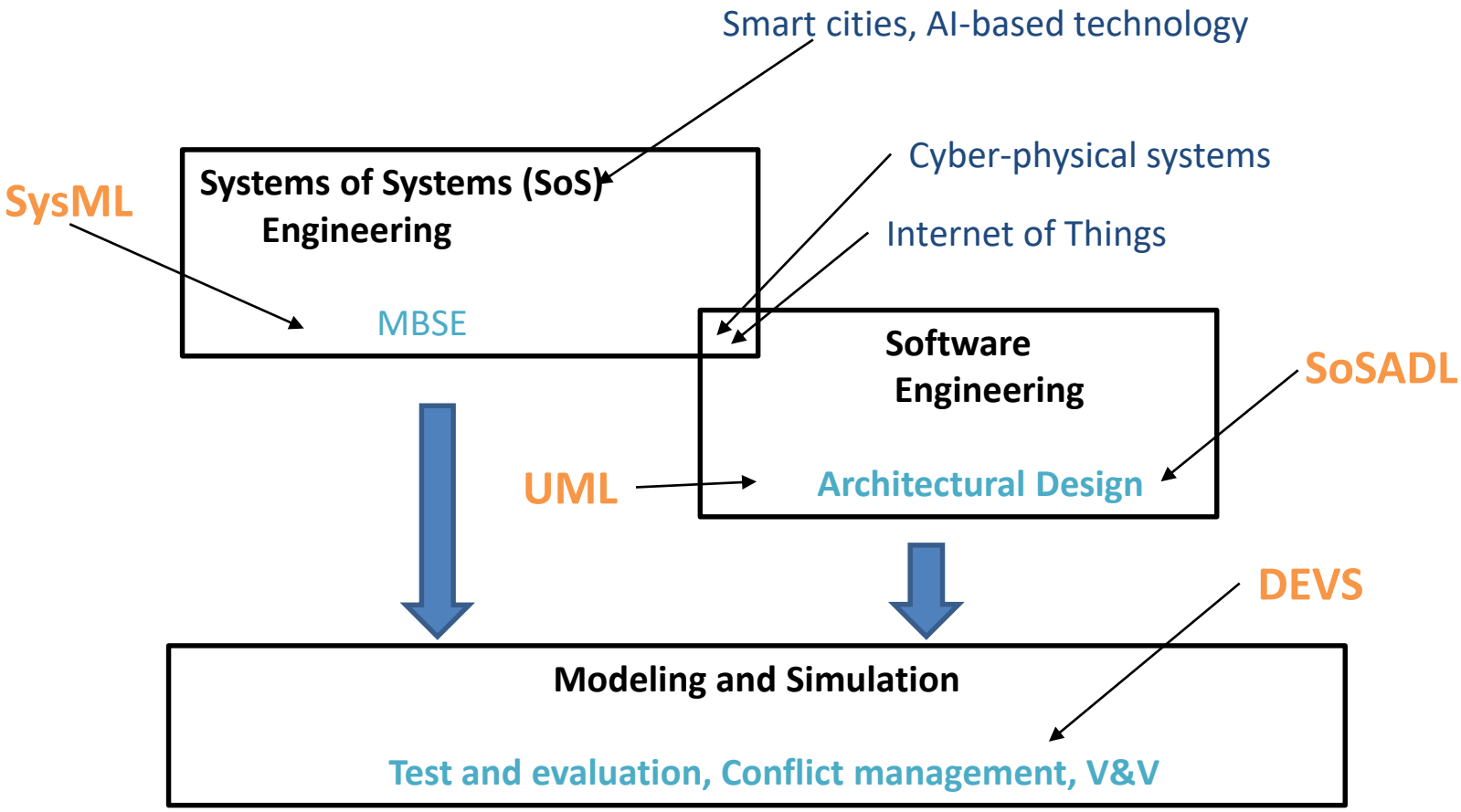
- Provides professional and student chapters top technology leaders and innovators from every sector of the computing industry.
- New speakers and new topics are added on an ongoing basis.



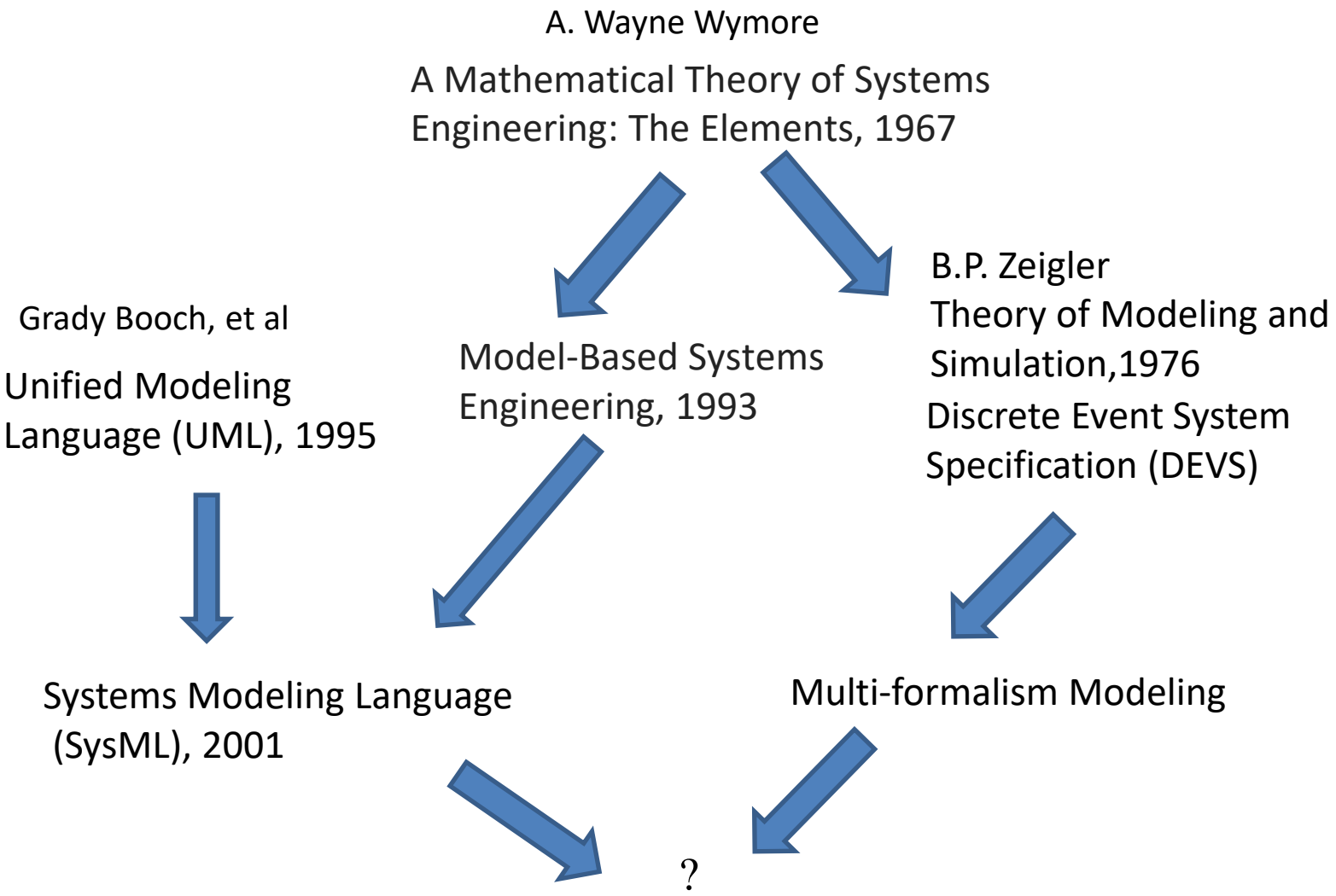
Modeling and Simulation in support of Systems and Software Engineering



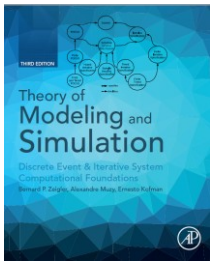
Modeling and Simulation in support of Systems and Software Engineering



History of MBSE and DEVS

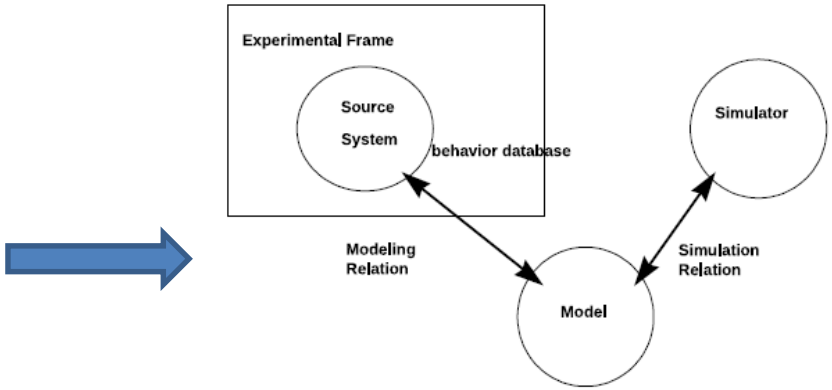


Theory of Modeling and Simulation



Levels of System Specification

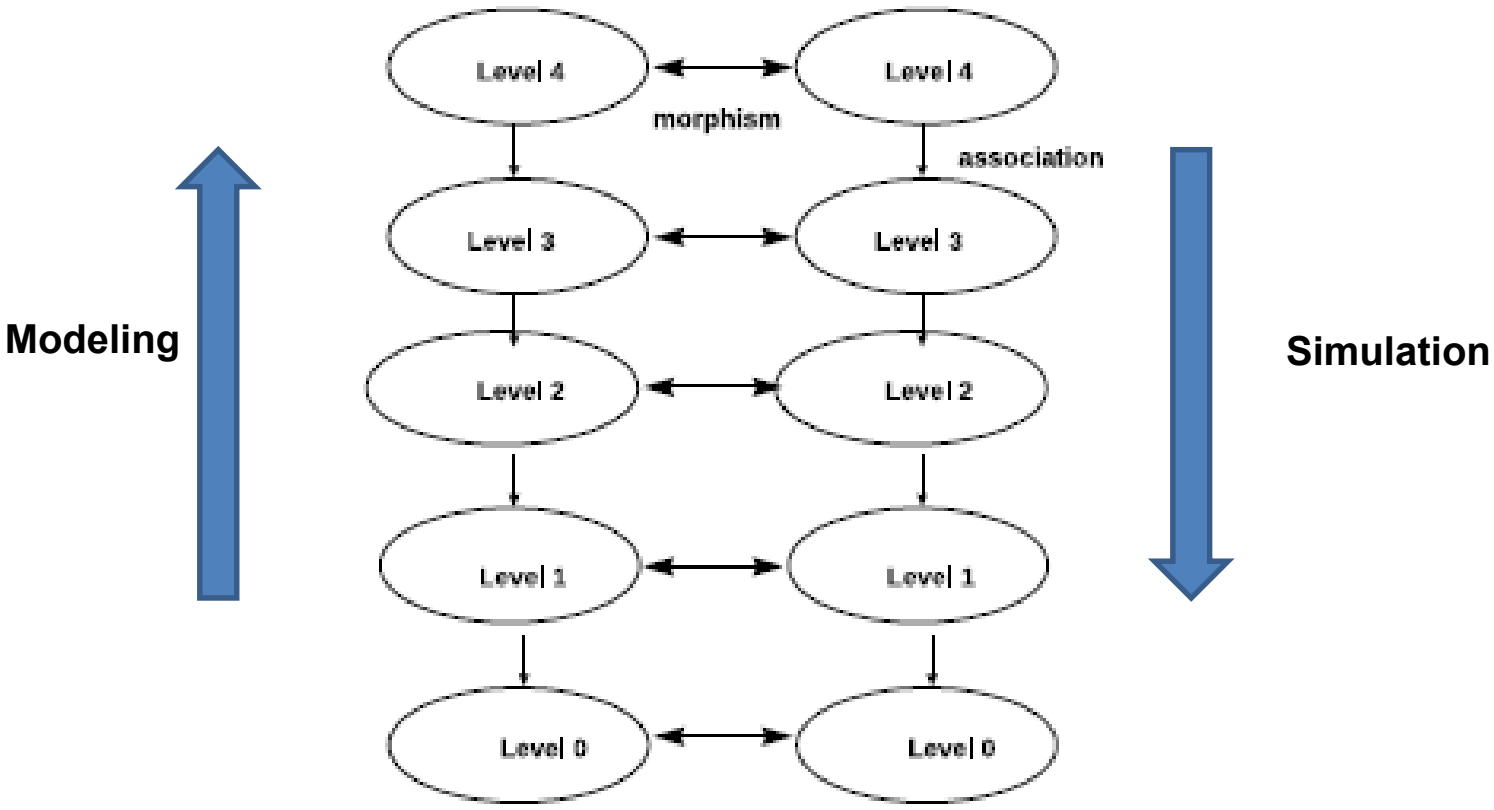
Level	Specification Name	What we know at this level	Example: A Person in a Conversation
0	Observation Frame	How to stimulate the system with inputs; what variables to measure and how to observe them over a time base;	The person has inputs and outputs at the usual cognitive level, such as streams of words
1	I/O Behavior	Time-indexed data collected from a source system; consists of input/output pairs	For each input that the person recognizes, the set of possible outputs that the person can produce
2	I/O Function	Knowledge of initial state; given an initial state, every input stimulus produces a unique output.	Assuming knowledge of the person's initial state when starting the conversation, the unique output response to each input.
3	State Transition	How states are affected by inputs; given a state and an input what is the state after the input stimulus is over; what output event is generated by a state.	How the person transits from state to state under input words and generates output words from the current state
4	Coupled Component	Components and how they are coupled together. The components can be specified at lower levels or can even be structure systems themselves – leading to hierarchical structure.	A description of a person's I/O behavior in terms of neural components and their interaction by spikes is at this level.



Basic Entities in M&S

Basic Entity	Definition	Example: A Person in a Conversation
source system	real or artificial source of data	Participants' in a conversation
behavior database	collection of gathered data	I/O Behavior as in levels of system specification
experimental frame	specifies the conditions under which system is observed or experimented with	Observation of participants' stream of words in a conversation
model	instructions for generating data	Coupled model of Finite state generator and recognizer implemented in neural form
simulator	computational device for generating behavior of the model	Discrete Event Simulation Environment

Equivalences at each Level of System Specification



DEVS Model of Spiking Neural Net

Atomic Model: Leaky Integrate and Fire (LIF) Neuron

<X,S,Y,δ_{int},δ_{ext},λ,ta>

X={p}
S={0,1,2,..,Th}
Y = {p}

$\delta_{ext}(s < Th,p) = s+1$

$\delta_{int}(Th) = 0$, $\delta_{int}(s>0) = s-1$, $\delta_{int}(s=0) = 0$

ta(T)=tFire, ta(s>0) = tLeak, ta(s=0)=∞
λ(s>0)=p, λ(s=0)=ϕ

Coupled Model: Neural Net

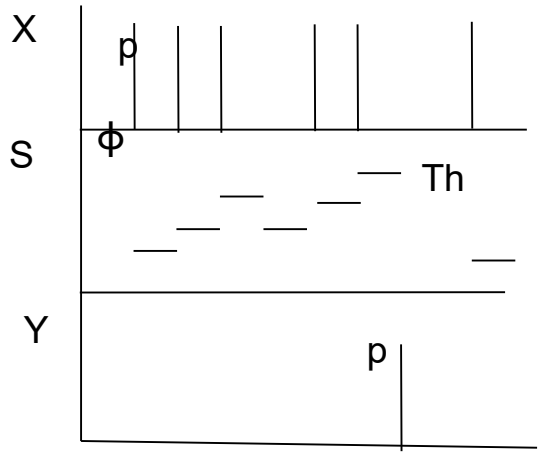
Internal couplings:

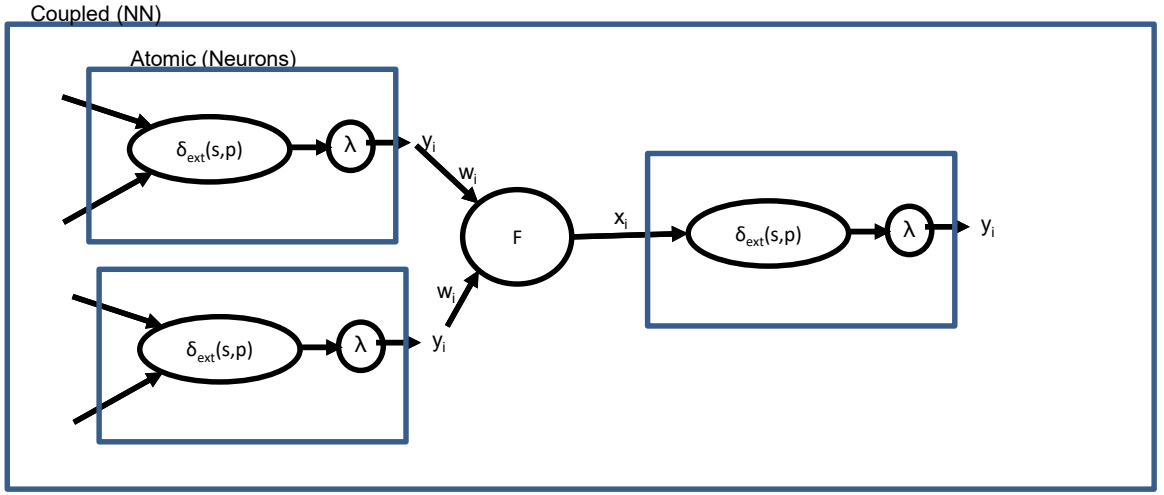
I_d is the set of influencers of d, $I_d \subseteq D$, $d \notin I_d$

$Z_j^{\otimes} : \prod_{i \in I} Y_i \rightarrow X_j$

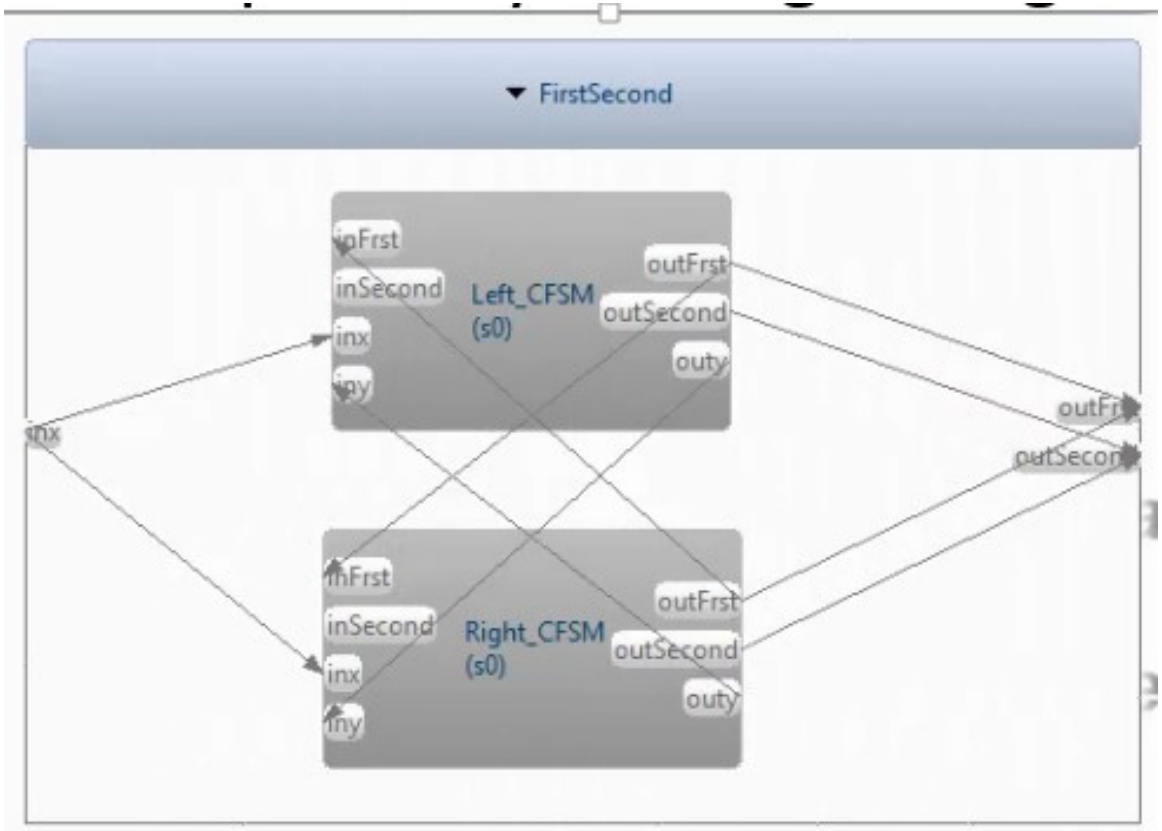
$Z_j(\ldots y_i \ldots) = F(\langle w_i * y_i \rangle)$

IO Behavior





A behavior required by self-organizing NN

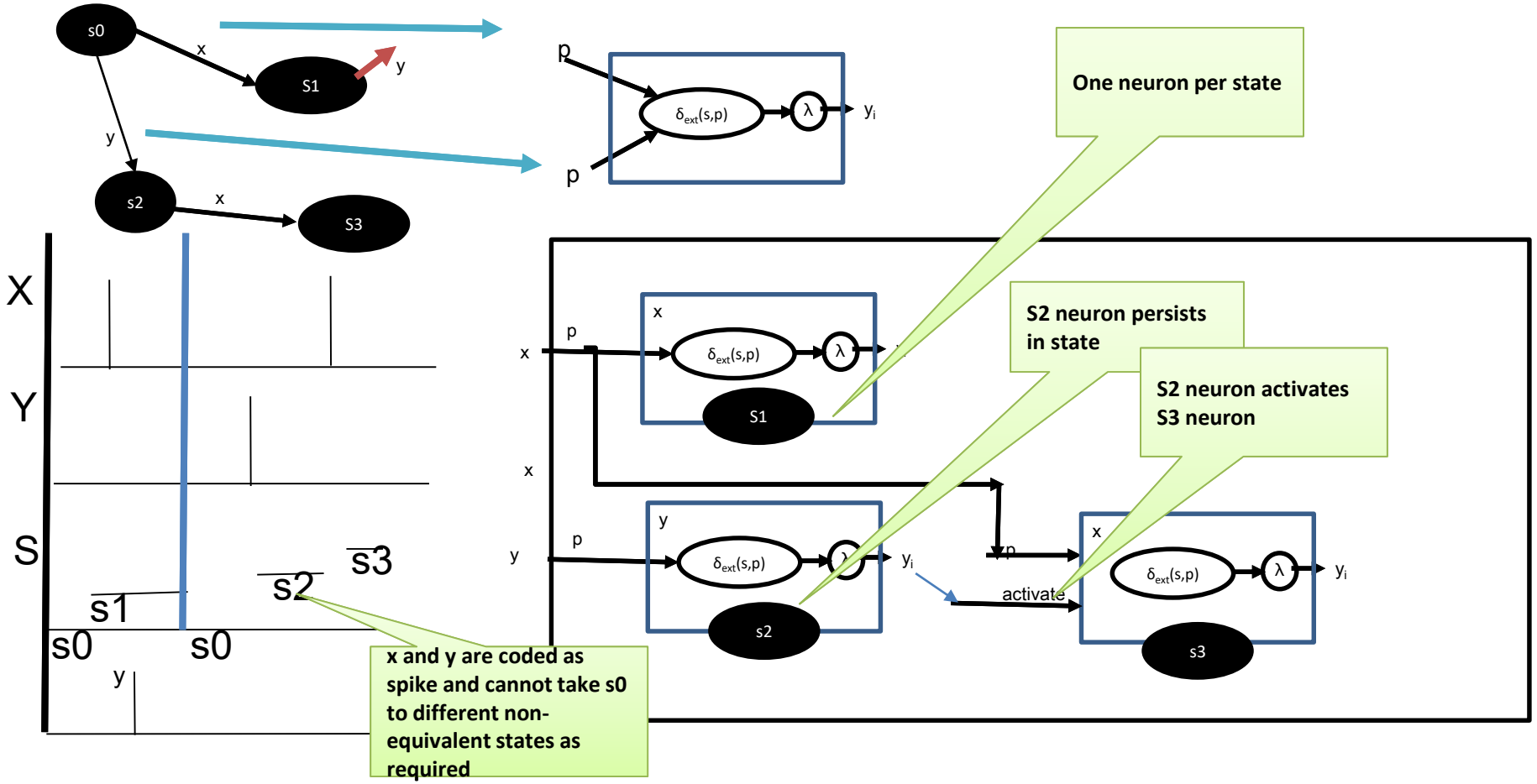


A behavior required by self-organizing NN

Left and Right cells have identical internal structure...
they are able to identify who receiv
the first and second inputs...
using signals and states

https://www.youtube.com/watch?v=UPCAqHP9GYo

A seemingly simple behavior that spiking LIF neuron model can't display – we need suitably configured higher level unit to implement it



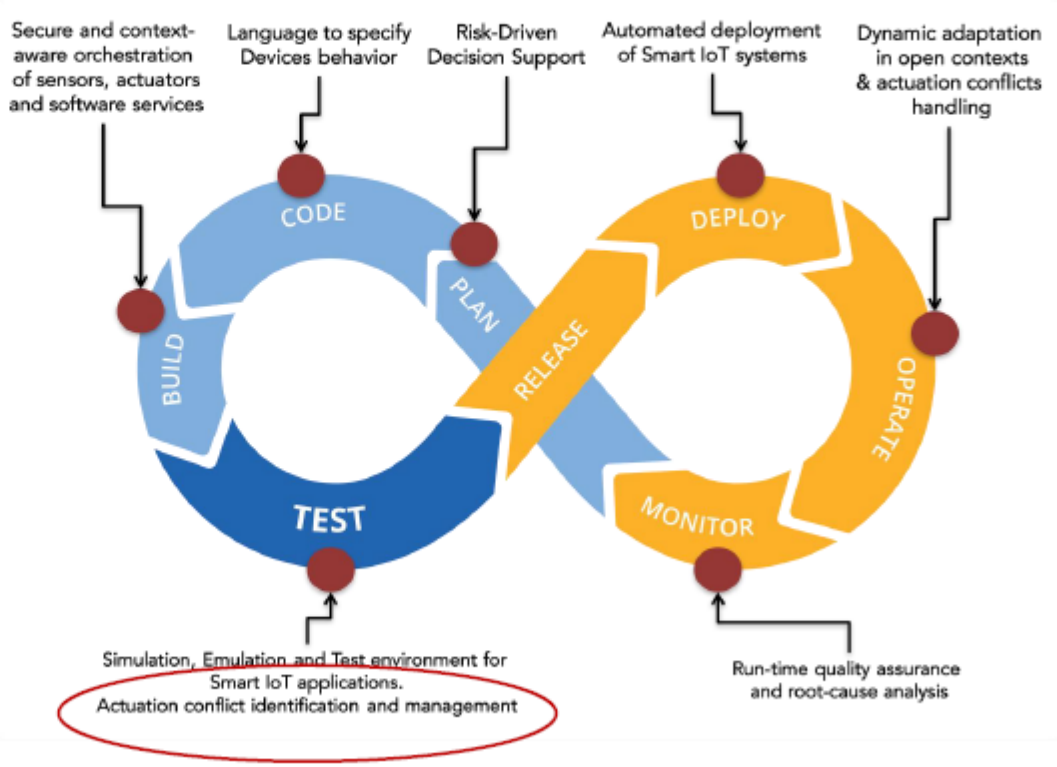
DEVS Systems-based Simulation is applicable to wide spectrum complex systems

- Theoretical – supporting application
  - Closure under coupling, universality, uniqueness, relation to other formalisms
  - Hierarchical Model Construction supports complex system development
  - Supports the correctness of the algorithms and validation of the executing models, e.g., time management is rigorously defined
  - Discrete Event System Specification (DEVS) formalism can be easily expanded beyond discrete-event world to continuous characteristics of system (Hybrid-DEVS)
- In Application
  - Models, Simulators and Experimental Frames are distinct entities with their own software representations
  - Precise and well-defined mathematical representation
  - Models/Experiments are developed systematically for interoperability
  - Repositories of models/experiments created and maintained systematically
  - Components can be easily reused for constructing new models
  - Discrete-event basis improves performance (e.g. no need for a global clock to control timing)
  - DEVS software can be deployed on distributed computing environments and interact with heterogeneous M&S system.

Application Example – European IOT Software Project

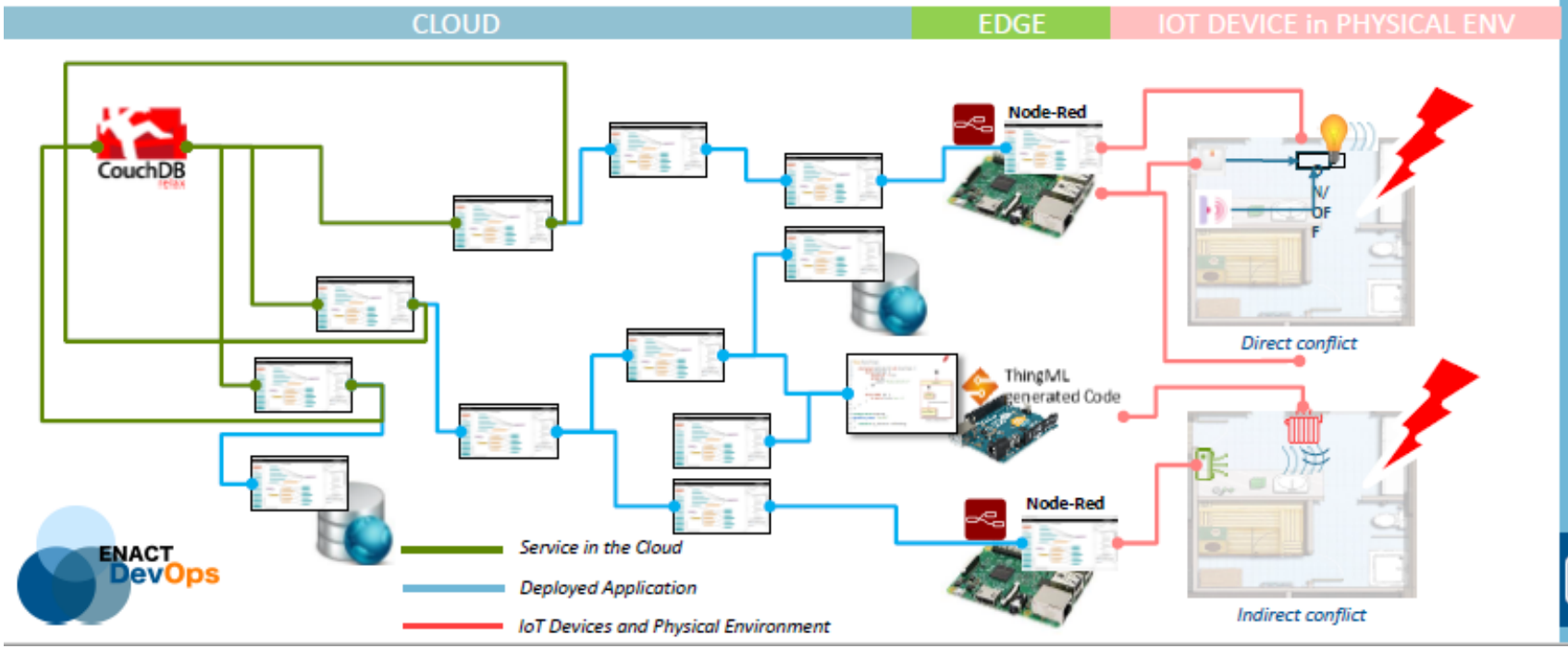
Approach

ENACT DevOps



Actuation Conflict Management in large scale

- **Application before Actuation Conflict Management**



Design of reliable Actuation Conflict Managers in a specific case and DEVS

- **Safe Actuation Conflict Manager Assisted Design**



ENACT DevOps

www.enact-project.eu

MBSE and DEVS: Development and Test of New Message Set for Tactical Data Link Standard



Requirements for new Protocol message set



Distributed Interactive Simulation (DIS)/Standard Interface for Multiple Platform Link Evaluation (SIMPLE

Design using UML/SysML State charts, Class diagrams and Interaction diagrams

Translation into DEVS and SES equivalents

Simulation based testing in net-centric environment

standards conformance testing

Rational Rhapsody Development Environment

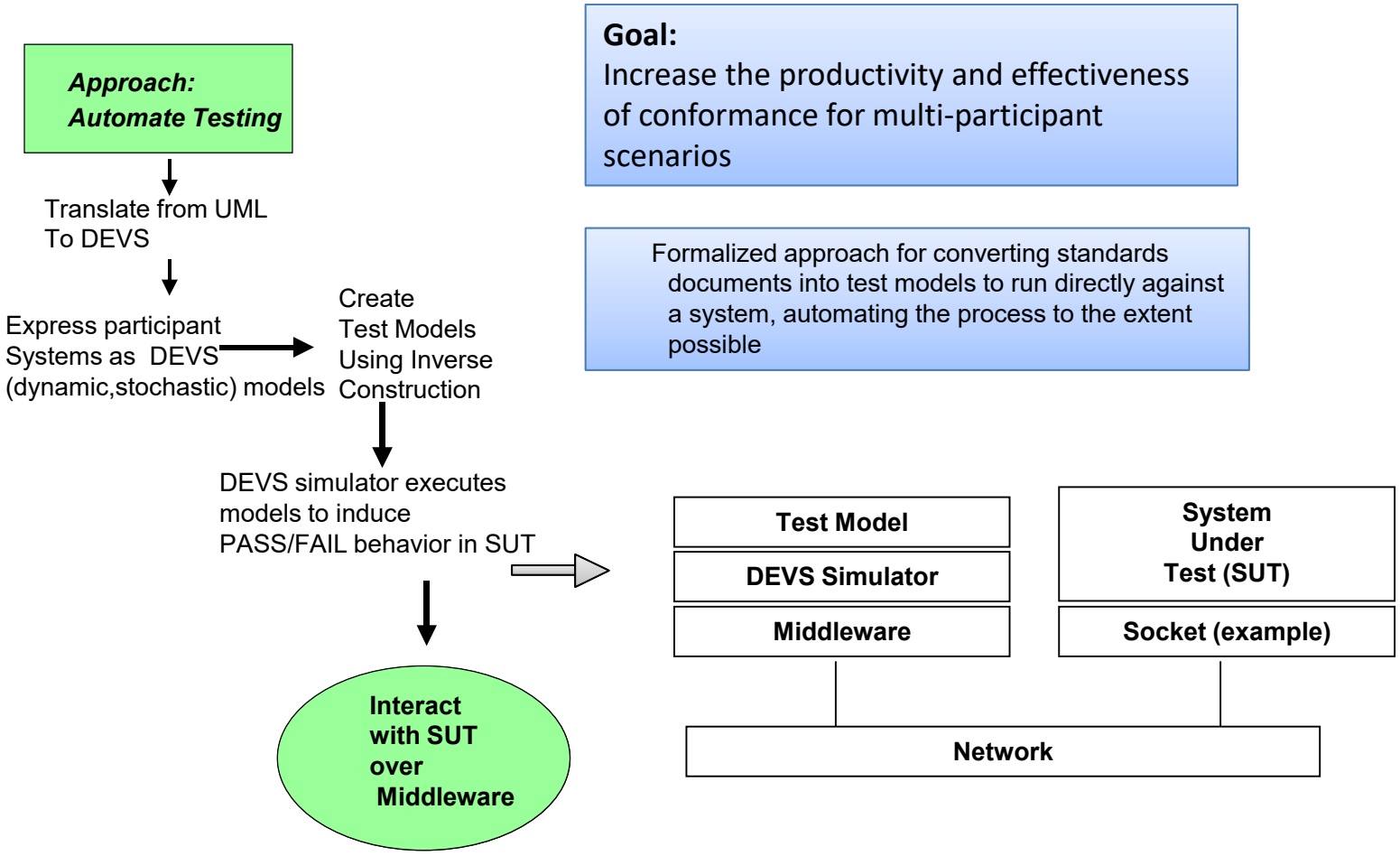
MS4 Me DEVS IDE



Semi-automated test suite design using inverse modeling methods

Remote testing of implementations over the Web

Automated Testing of Message Protocol Standards



Goal:
Increase the productivity and effectiveness of conformance for multi-participant scenarios

Formalized approach for converting standards documents into test models to run directly against a system, automating the process to the extent possible

Message Protocol Standard Specification

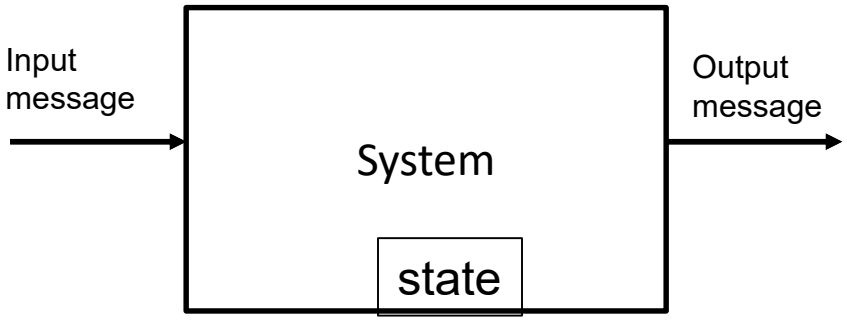


Document

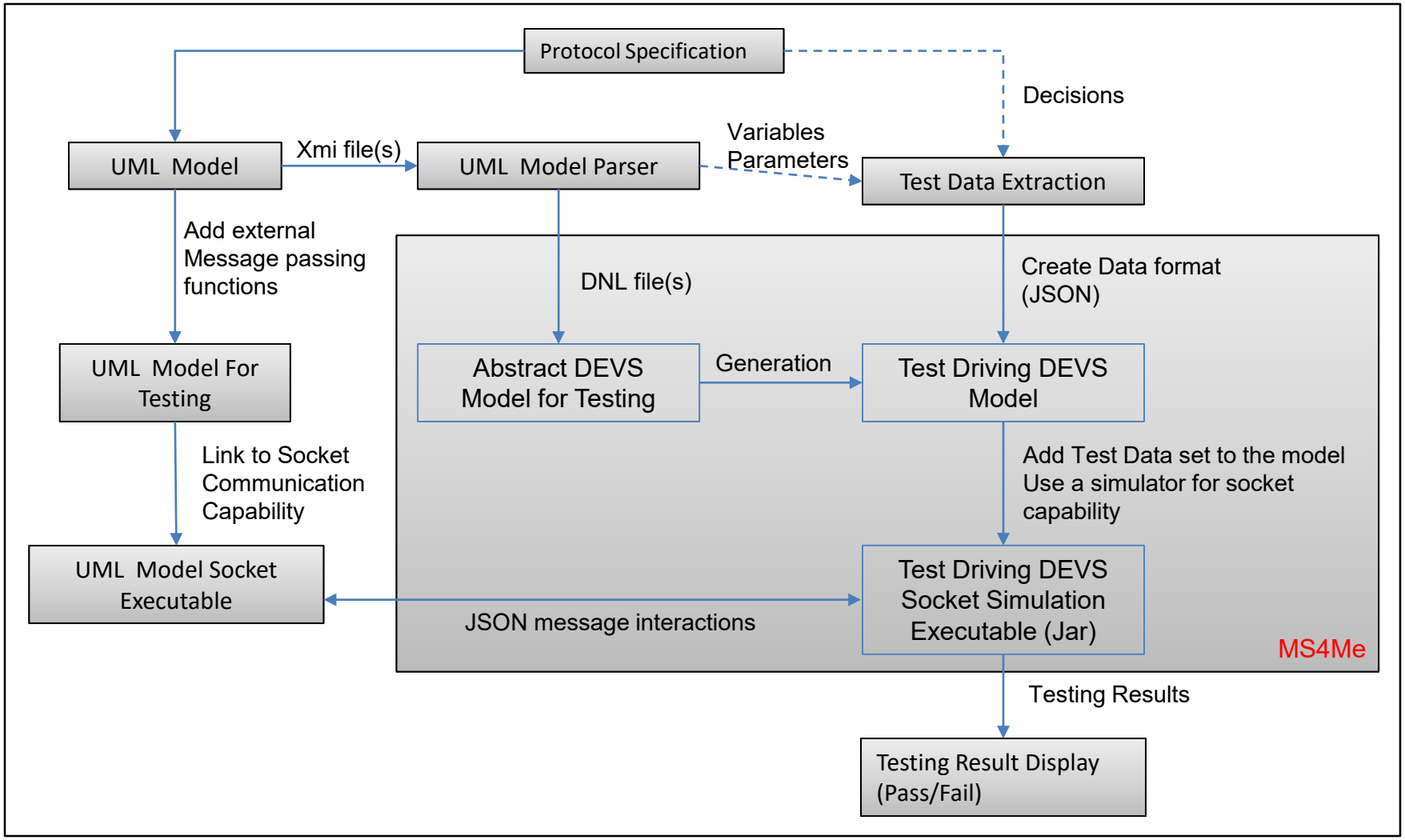
Prescribes

IO Pair Name	Message Name	State Name	Specification
1		Prepare	
2	DoTask	TransmitTask	
3	ReceivedTask	Acknowledge	
4		PerformTask	

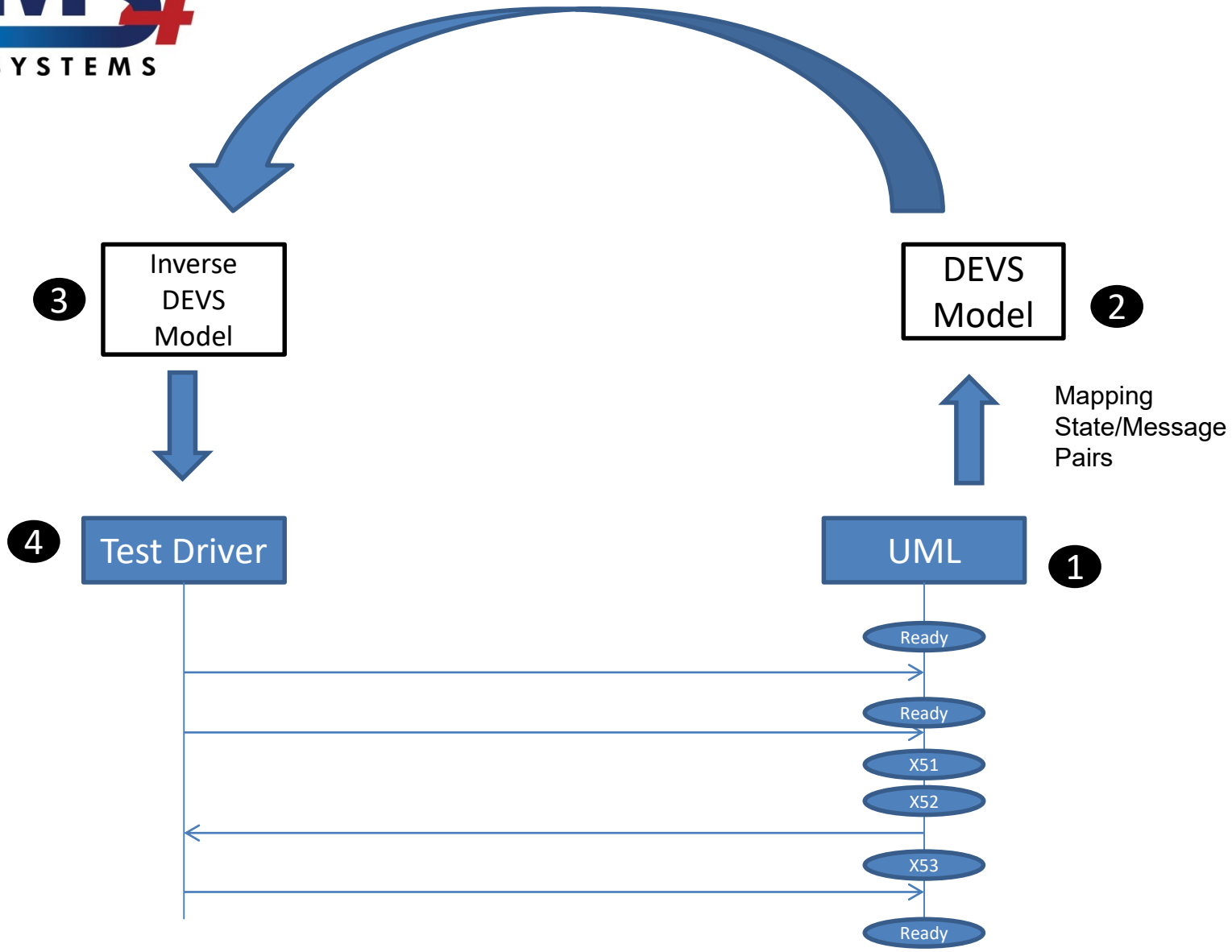
Messages can be:
- Queries
- Commands
- Responses
- Information



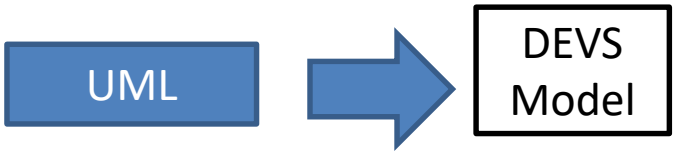
Overall Architecture of UML to DEVS Model Testing Methodology



DEVS-Based Test Development



Mapping State/Message pairs to DEVS

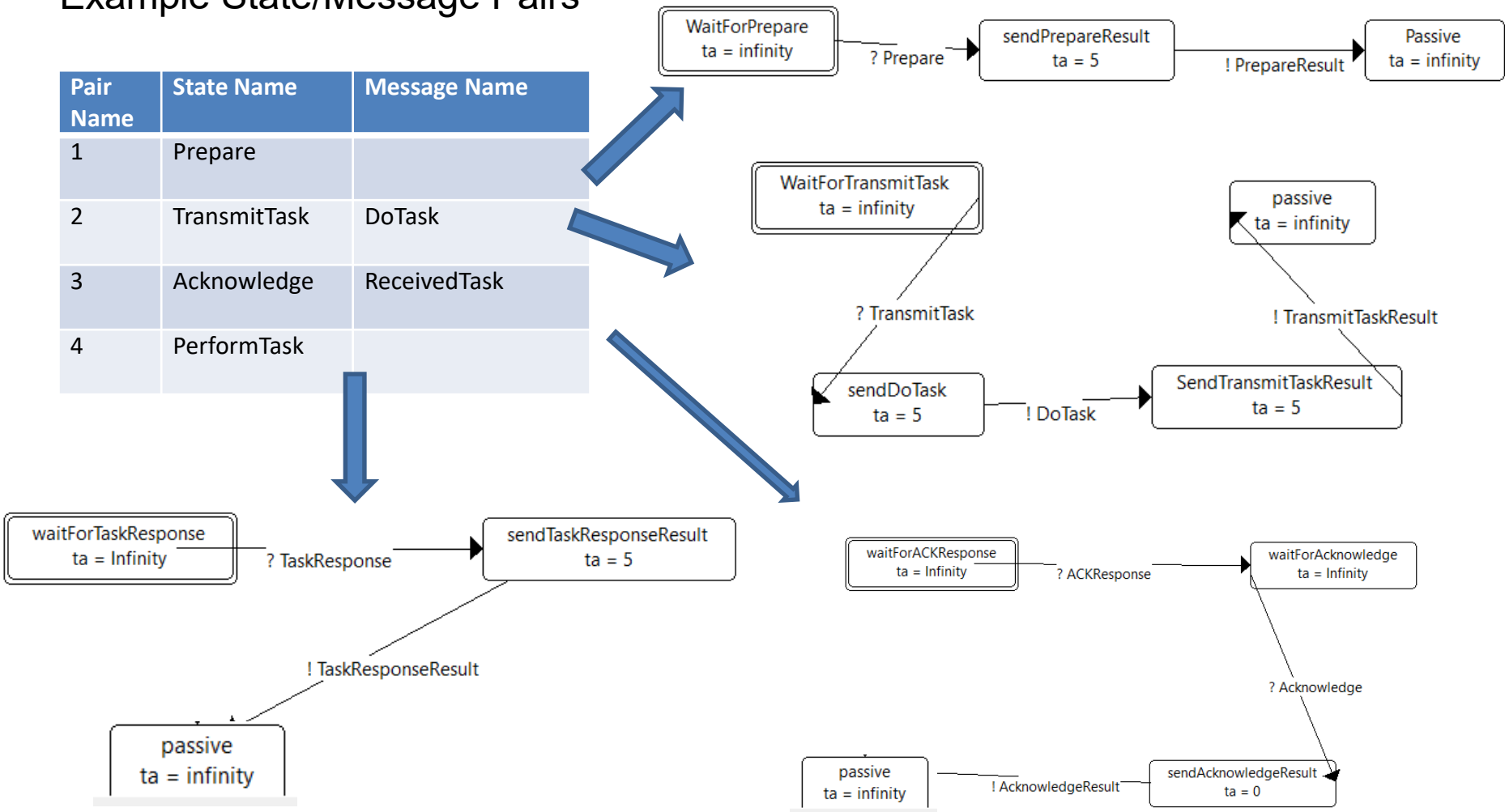


Transaction	state	message	Input		Output	
			State	Input message	State	Output message
Preparation	X		"waitFor" + X	"?"+ X	"send"+X (transaction number) +"_Result"	"!"+ X(transaction number) +"_Result"
Transmission	X	M	waitFor" + X	"?"+ X	"send"+M "send"+ X(transaction number) +"_Result"	"!"+M+"_Msg", "!"+ X(transaction number) +"_Result"
Reception 1	X	M	"waitFor" + M waitFor" + X	"?"+ M+ "_Msg" "?"+ X	"send"+ X(transaction number) +"_Result"	"!"+ X(transaction number) +"_Result"
Reception 2	X		waitFor" + X	"?"+ X	send+X (transaction number) +"_Result"	"!"+ X(transaction number) +"_Result"

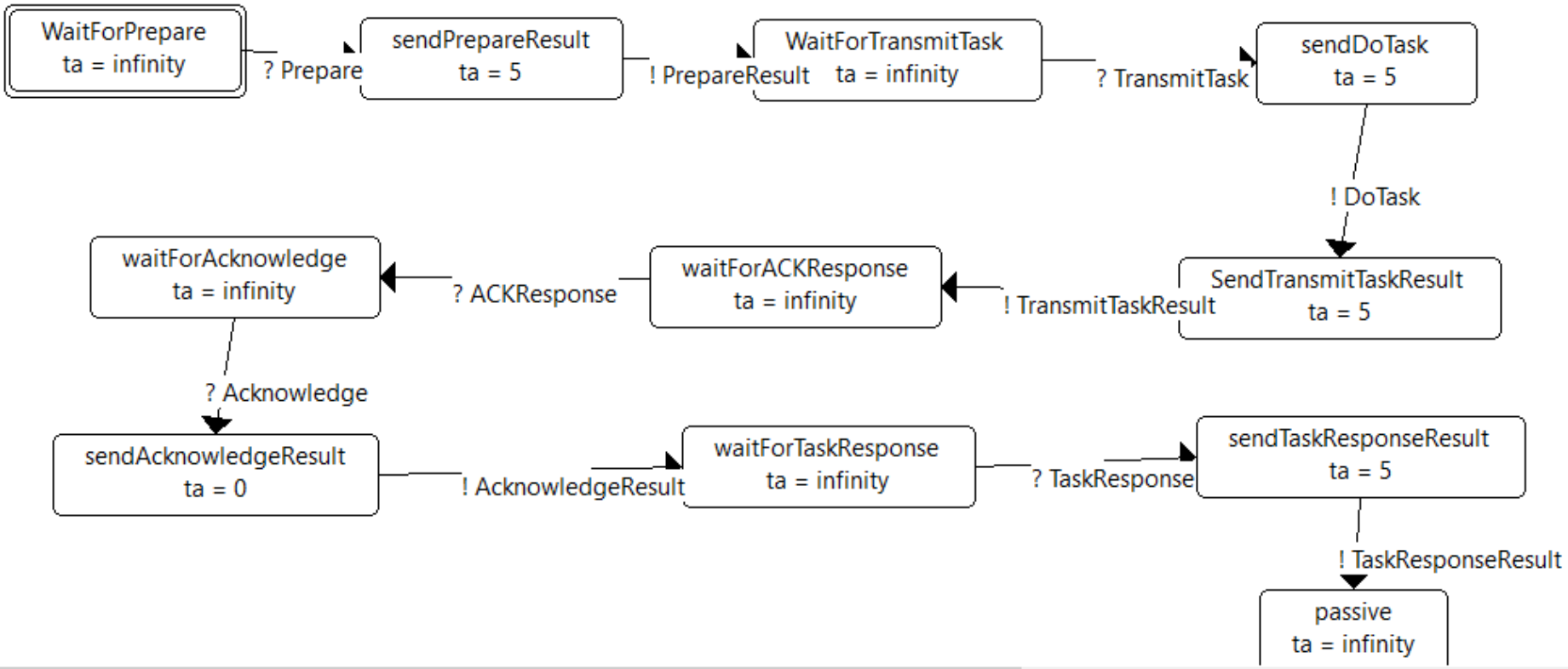
DEVS models Derived From UML Specification

Example State/Message Pairs

Pair Name	State Name	Message Name
1	Prepare	
2	TransmitTask	DoTask
3	Acknowledge	ReceivedTask
4	PerformTask	



Composed DEVS Model

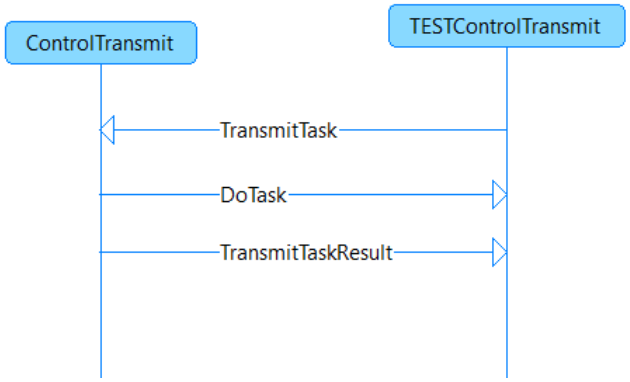


Inverse DEVS Model becomes Test Driver



Conversion	Abstract DEVS model (from)	Inverse DEVS Test Model(to)
State	state	TESTstate
TA	Infinity	5
	Not Infinity	Infinity
Message	Output Message	Input Message
	Input Message	Output Message

Running Test Driving DEVS Model




```
[80.0, 9:35:54.28] Testing Finished !
[80.0, 9:35:54.28] Passed : 3 out of total test data : 4
[80.0, 9:35:54.28] Testing Result : Passed rate : 75.0%
```

Simulation output

Run a coupled model with PES

```
{
"MsgType":"DoTask",

"Argument1":2,

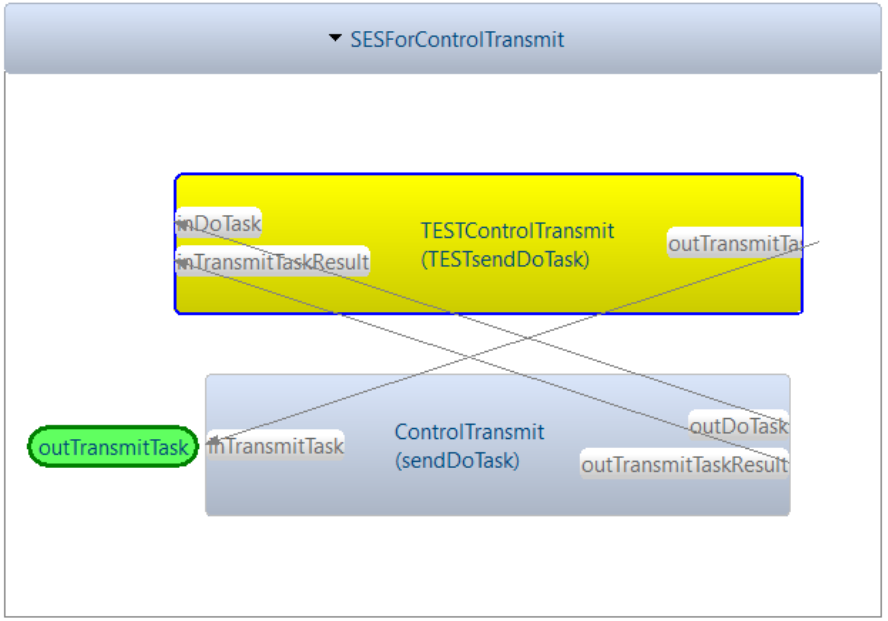
"Argument2":"11",

"Argument3 ":"10"

}
```

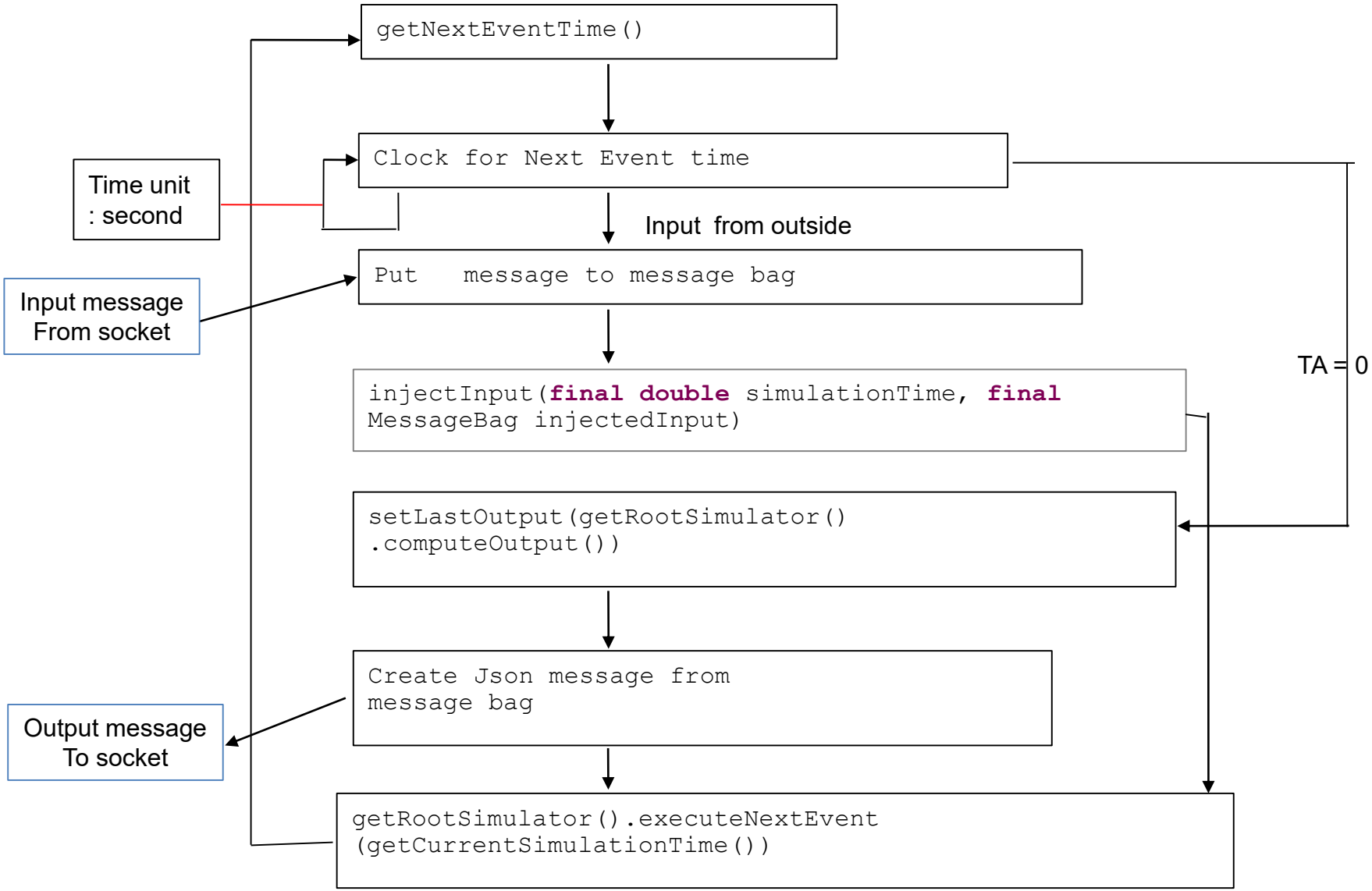
Input Message
Output Message

DoTask.java
- public void setJson(String jsonMsg)
- public String toJson()



MS4Me automatically generates toJson and setJson functions for DEVS messages

Distributed Simulation Protocol

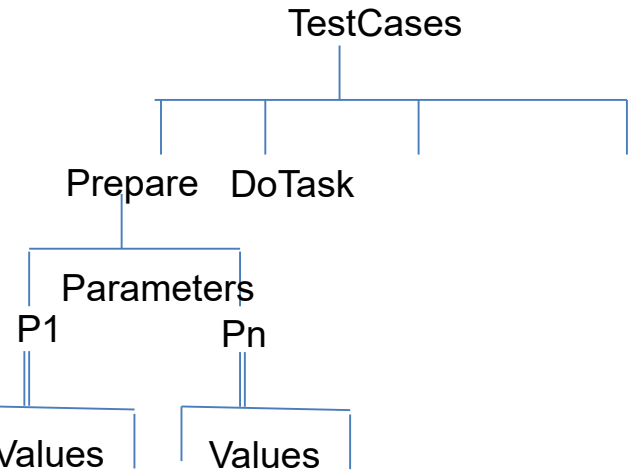


Operational Test Driver Environment

Pair Name	State Name	Message Name
1	Prepare	
2	TransmitTask	DoTask
3	Acknowledge	ReceivedTask
4	PerformTask	

System Entity Structure

SES



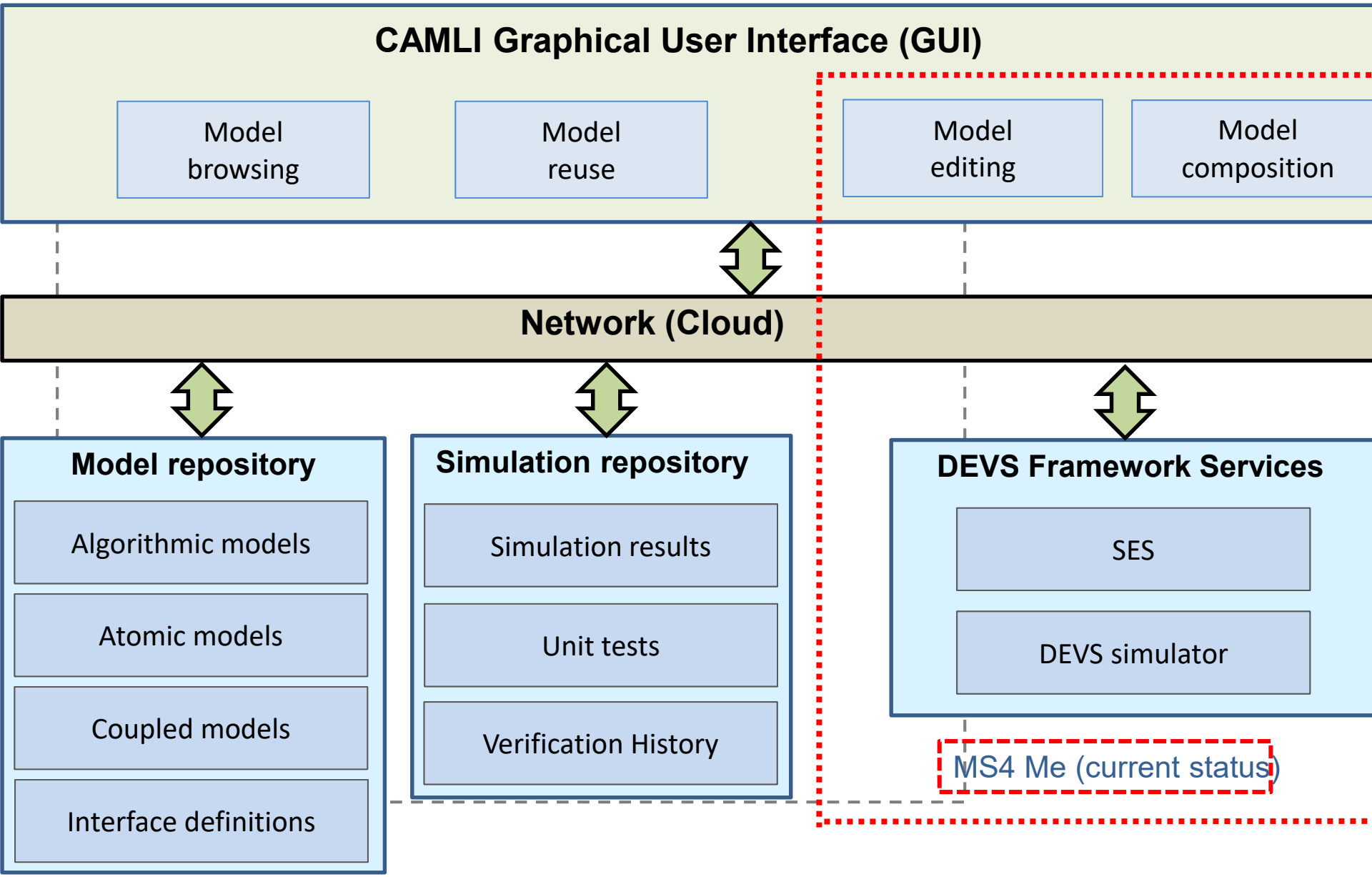
Pruning & generation



RTSync's Projects illustrate DEVS Applications

- Model–Level Integrated Simulation Architecture for Collaborative Development
- Automated Selection/Construction of Scenarios for Multi-missile engagements
- Methodology for automated test of data link protocols
- Generation of labelled training data for deep learning of radar signatures
- Value-based Pathways-based Coordination of health care

Model–Level Integrated Simulation Architecture for Collaborative Development

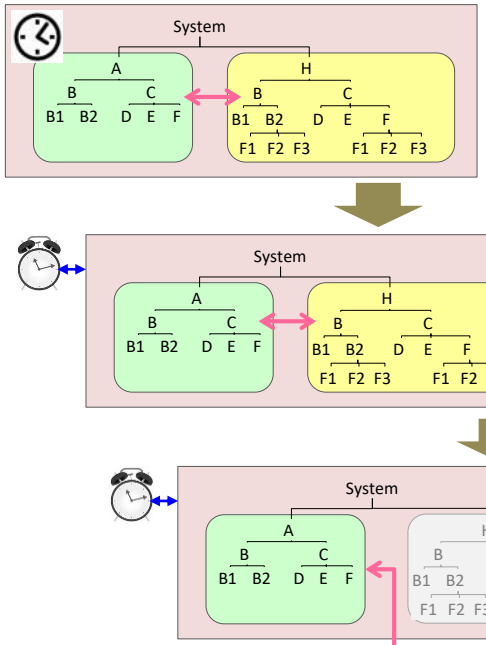


Model continuity: Transform models for real time application

- Same DEVS models can be designed, evaluated and transformed to operating system for actual real application.

Monitor & Control System
(ex.. SCADA)

System under test
(ex.. Plant, Power System)



Phase 1: Simulation by simulation clock

- Simulate models and SES with simulation clock
- Evaluate various design specifications and performance

Phase 2 : Simulation by real-time clock

- Continue to use same models of phase 1
- Run simulation with real-time clock
- Evaluate real-time characteristics of system

Phase 3 : Transform models to real application

- Continue to use same models of phase 2
- Directly interface to real world application



Books :

“Model Engineering for Simulation”, Editors: L. Zhang, B. P. Zeigler, and L. Lian

“Theory of Modeling and Simulation Third Edition”, B. P. Zeigler. Alexandre Muzy, E. Kofman, 2018

“Guide to Modeling and Simulation”, B. P. Zeigler and H. Sarjoughian , 2nd Edition, 2017

“Modeling and Simulation-based Data Engineering Introducing Pragmatics into Ontologies for Net-Centric Information Exchange”, B. P. Zeigler and P. Hammonds, 2007

Contacts

For more information , contact info@rtsync.com or POCs ;

Dr. Bernard P. Zeigler	Chief Scientist	zeigler@rtsync.com
Dr. Doohwan Kim	President & CEO	dhkim@rtsync.com