Control and Decision Communication Across Heterogeneous Model Types

Hessam S. Sarjoughian
Chao Zhang
Xuanli Lin

hss@asu.edu | http://sarjoughian.faculty.asu.edu/ | https://acims.asu.edu

MODPROD
Session 3b: Cyber-Physical Systems
February 3-4, 2021

ACIMS
ARIZONA CENTER FOR INTEGRATIVE MODELING AND SIMULATION

school of computing, informatics, & decision systems engineering

Ira A. Fulton Schools of Engineering
Arizona State University

observations on DEVS/Modelica co-simulation

- hybrid models are often needed for simulating system-of-systems, including Cyber-Physical Systems.
- co-Simulation uses some means for individual simulated models to interact
- co-simulating parallel DEVS with Functional Mockup Units
- co-simulation computation efficiency for the master DEVS-Suite and slave OpenModelica simulators

continuous and discrete dynamical models

model elements	system-theoretic modeling methods		
	Differential Equation	***Discrete-Time***	***Discrete-Event***
time base	continuous	discrete	continuous
inputs, outputs, and states	vector space	arbitrary	arbitrary
input segments	piecewise-continuous	sequences	discrete events
state and output segments	continuous	sequences	piecewise-constant

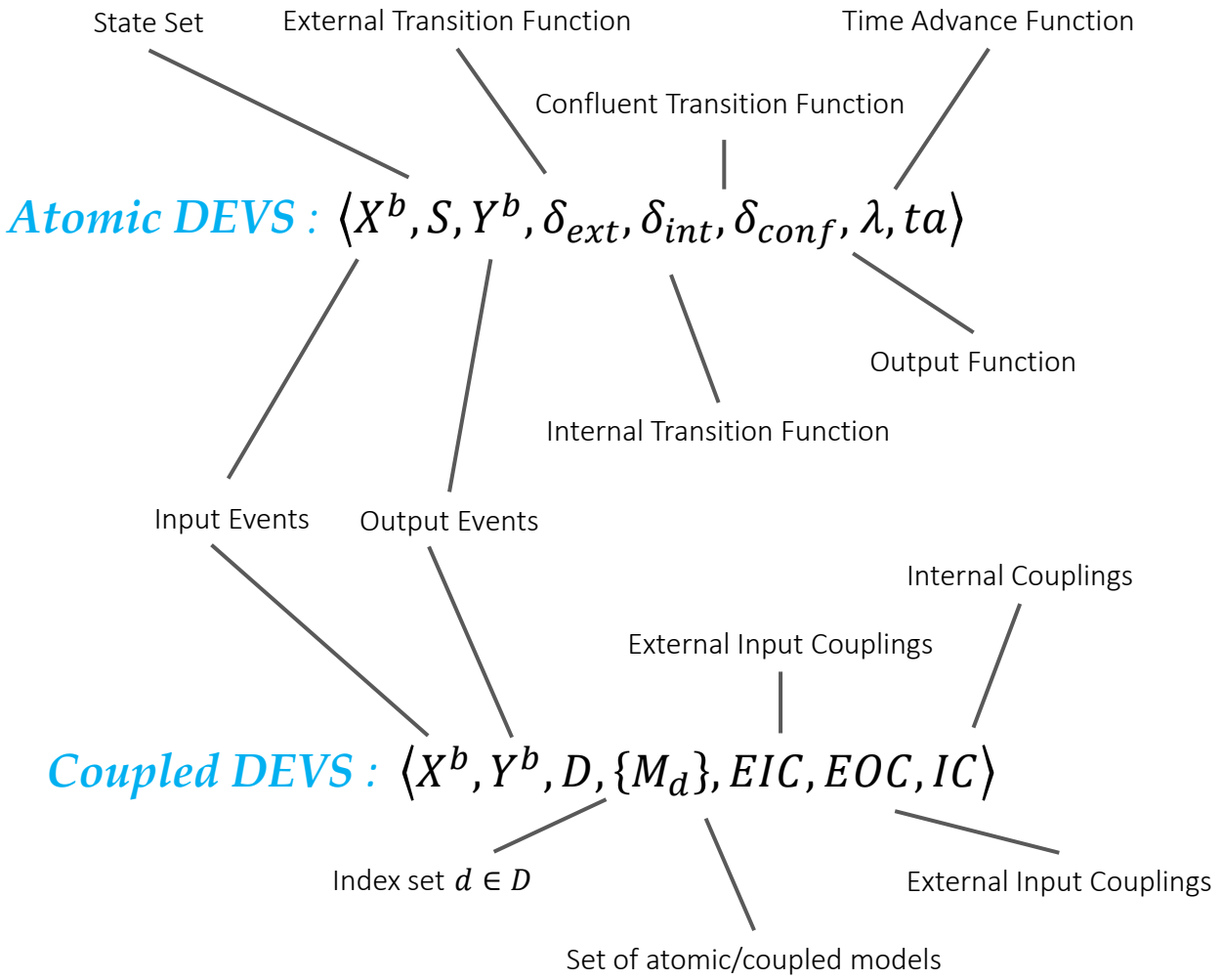
	DEVS	Modelica	DEVS-FMI Interface
data type	arbitrary	numeric	numeric
data trajectories	events/piecewise constant	piecewise constant	events

parallel DEVS formalism

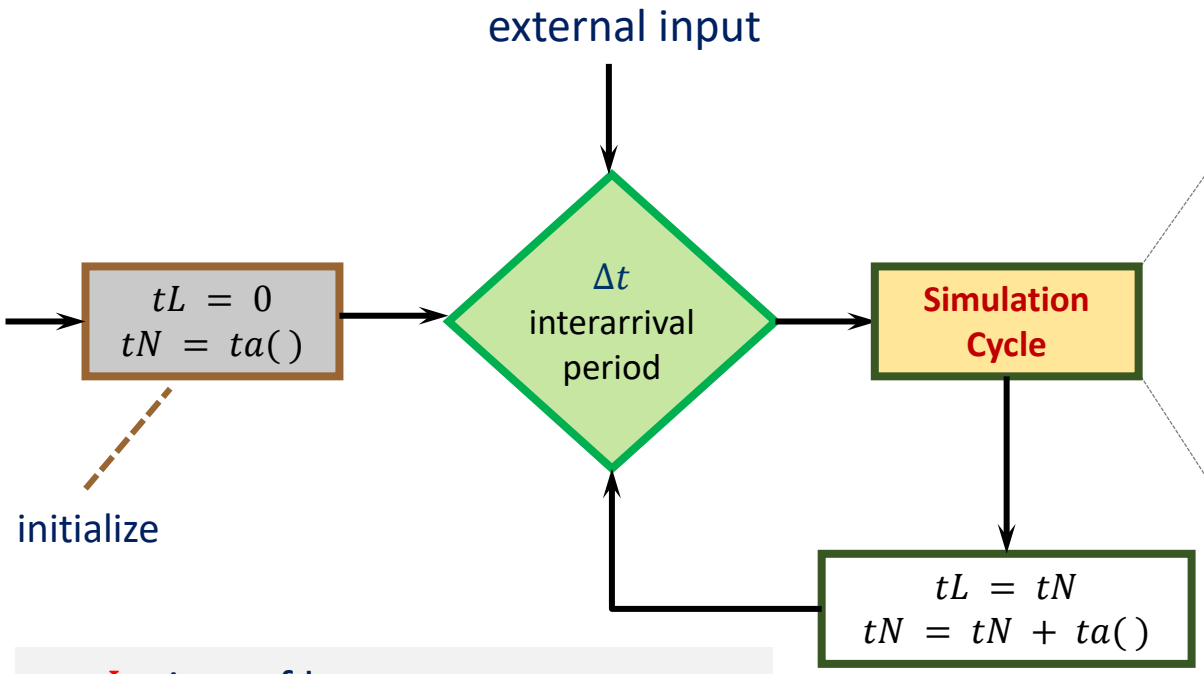
- system-theoretic
- strong I/O modularity
- strict composition hierarchy
- separate external & internal behaviors
- parallel behavior
- continuous time-base
- computationally efficient

atomic DEVS models define basic structure and state-time behaviors

coupled DEVS models define composition of atomic/coupled models



atomic DEVS model simulator

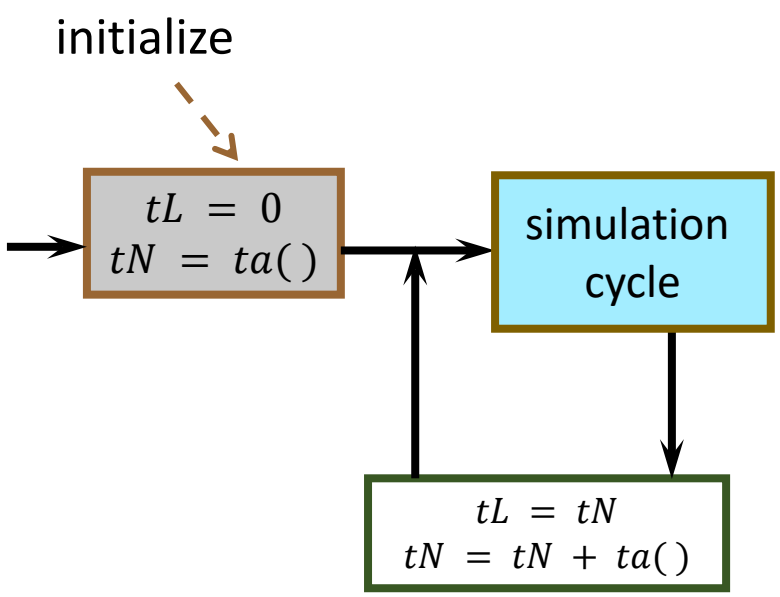


Execution protocol

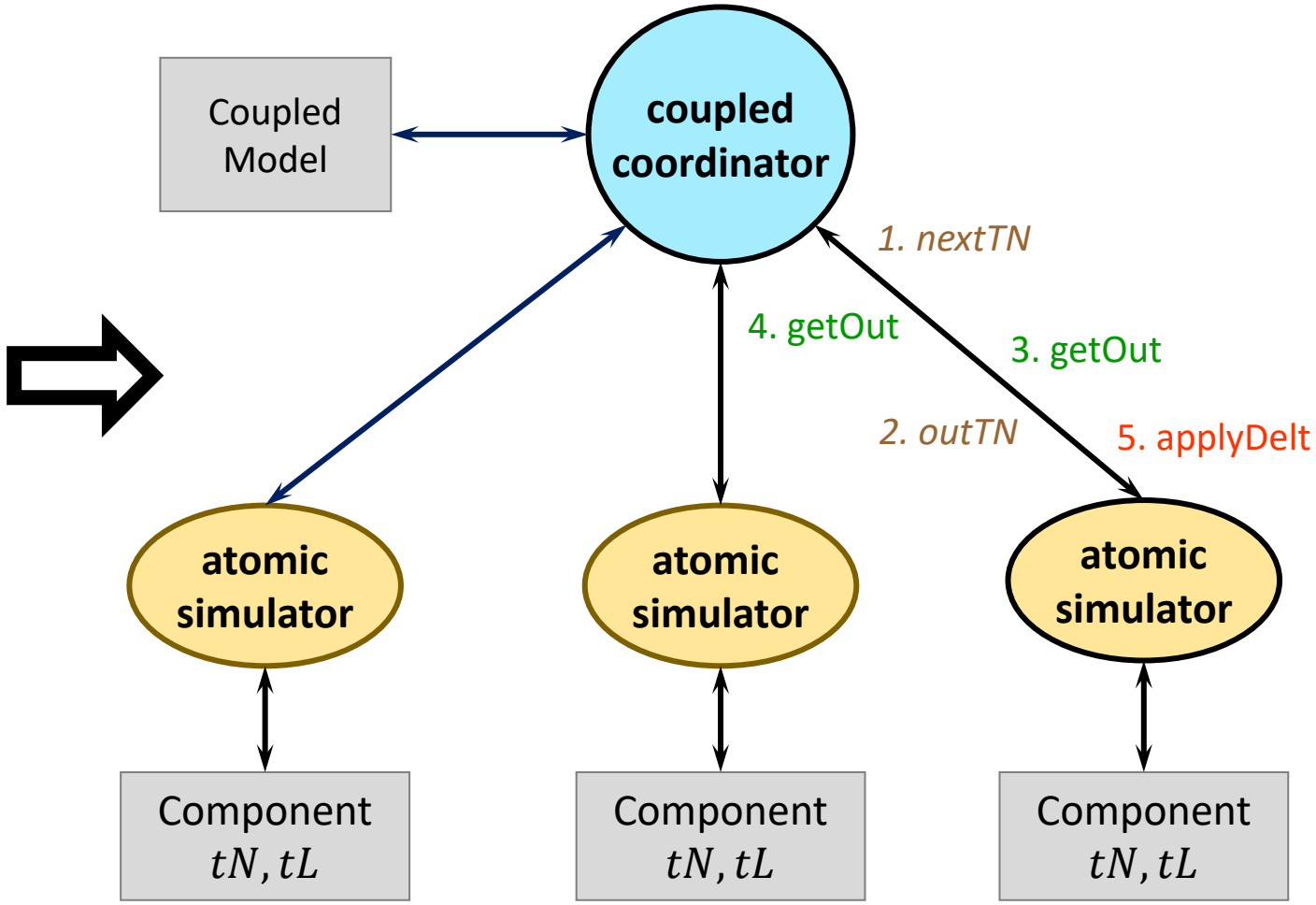
if no input event & no next internal state event
→ *do nothing*
else if any internal state event
→ *execute internal event*
else if any input event(s)
→ *execute input event(s)*
else if any internal state event ∧ any input event(s)
→ *execute input and internal state events in some order*

- tL: time of last event
- tN: time of next event
- Input can be injected at arbitrary interarrival periods $0 \leq \Delta t \leq \infty$

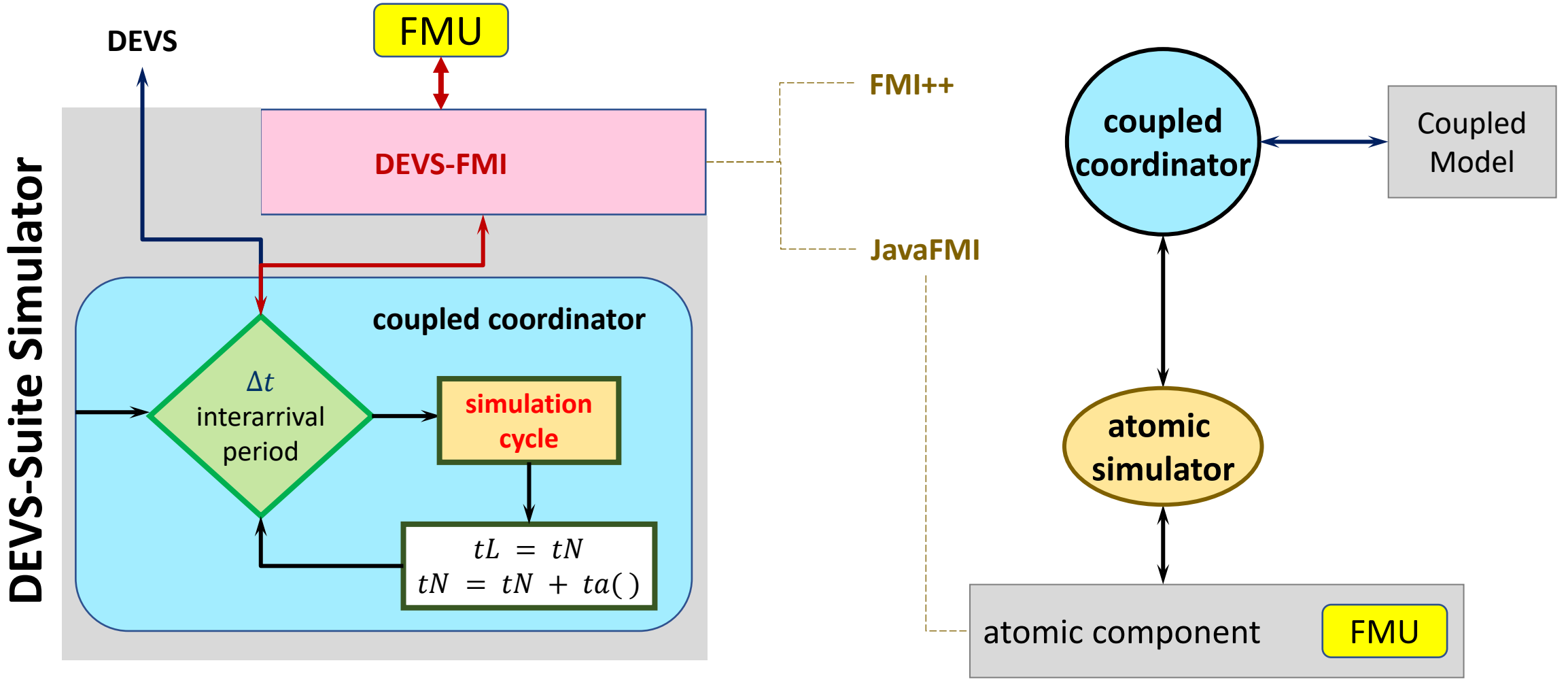
coupled DEVS model simulator



NOTE: *Component* can be a *Coupled Model* in which case *coordinator* would be needed instead of *simulator.*

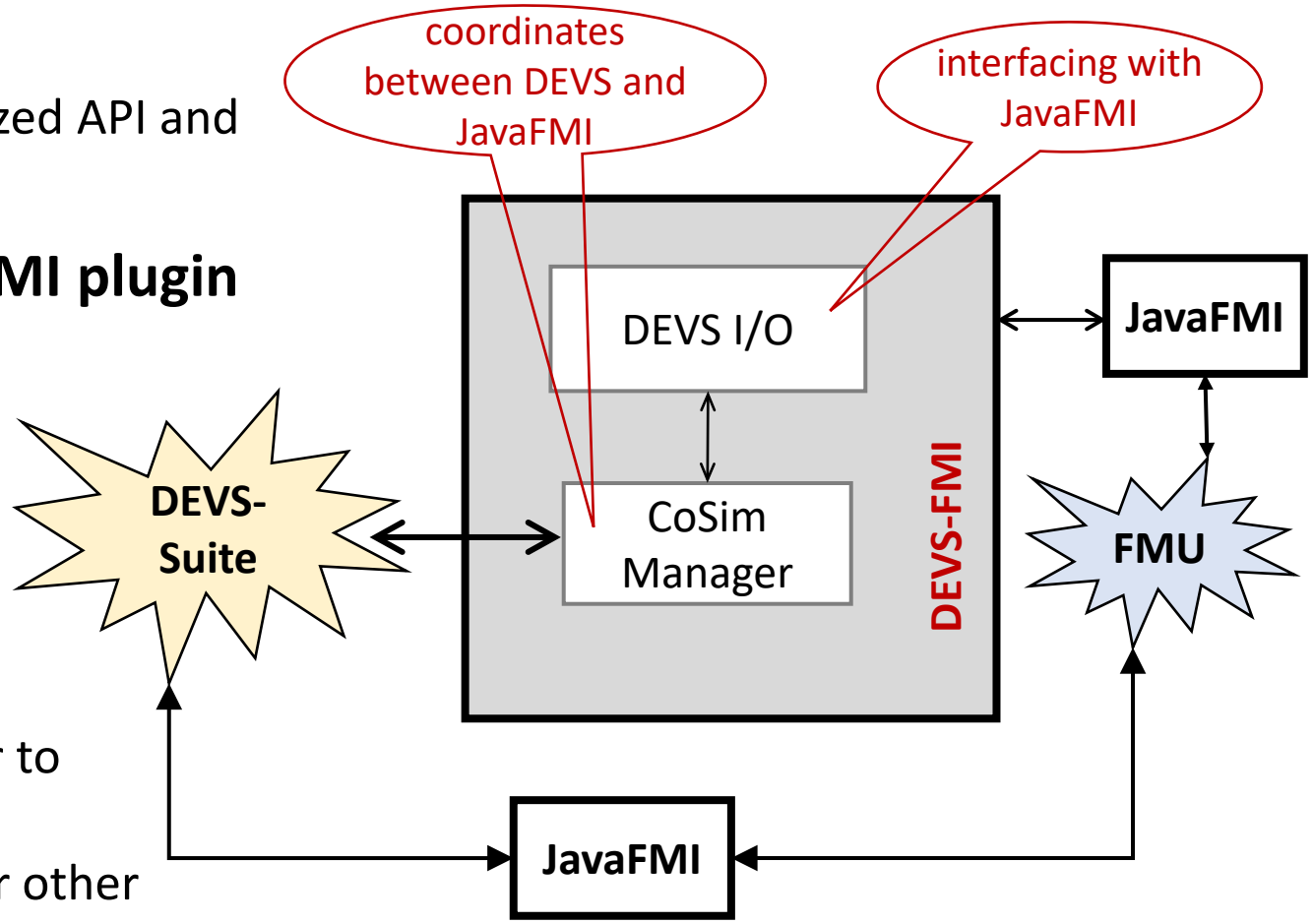


FMU execution: interfacing & embedding



FMI++, JavaFMI, and DEVS-FMI

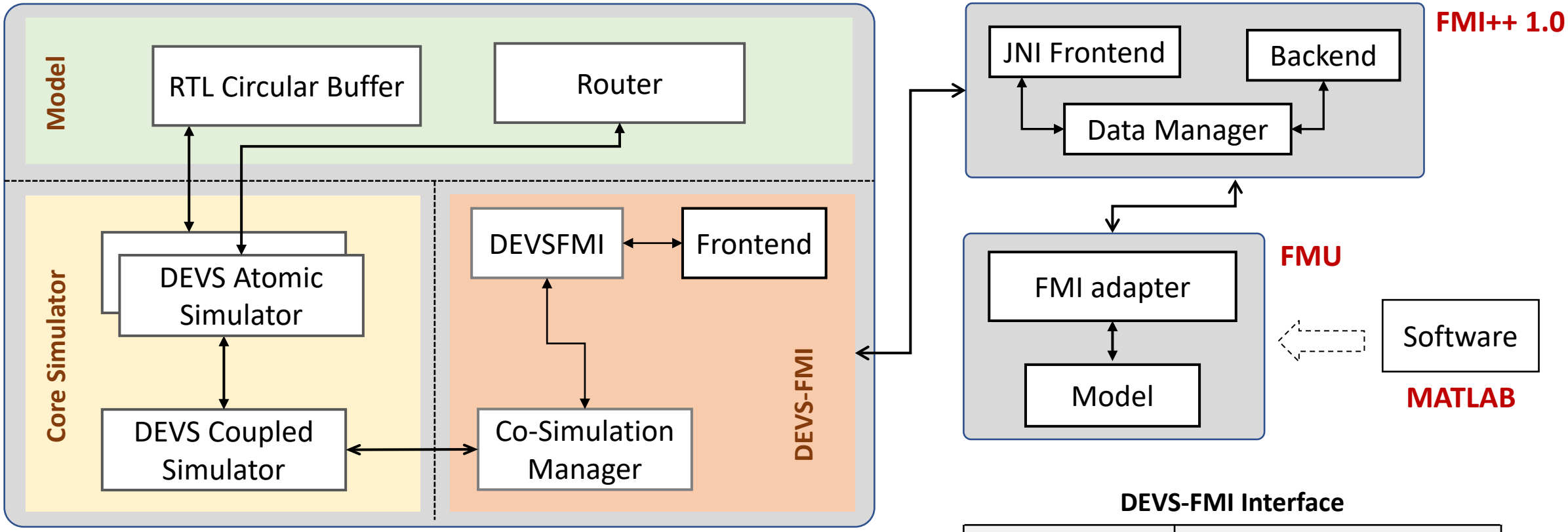
- **FMUs**
  - Modelica generates FMUs using standardized API and the actual model
- **DEVS-FMI (FMI++, JavaFMI) and JavaFMI plugin**
  - Start, initialize, terminate the simulation
  - Step the simulation
  - Cancel a step
  - Read/write to values from/to the FMU
- **DEVS-FMI**
  - Interacts with JavaFMI simulation wrapper to manage FMUs
  - Provides basic utilities that can be used for other co-simulations



DEVS-Suite and Matlab co-simulation

- Computer hardware + software
  - Circular Router Buffer (RTL-DEVS, DEVS) – master simulator
    - Circular Buffer
    - Router
  - Route finder (FMU/Matlab) – slave simulator
    - Finding routes in 2D Network-on-Chip – untimed
  - DEVS-FMI – co-simulator
    - Every FMU execution cycle is coordinated as black-box under the execution cycles of the coupled/atomic execution protocols
    - DEVS execution protocol synchronizes in lock-step the DEVS I/O with the FMU I/O at increasingly monotonic time instances.

co-simulator: DEVS-Suite, DEVS-FMI (FMI++)

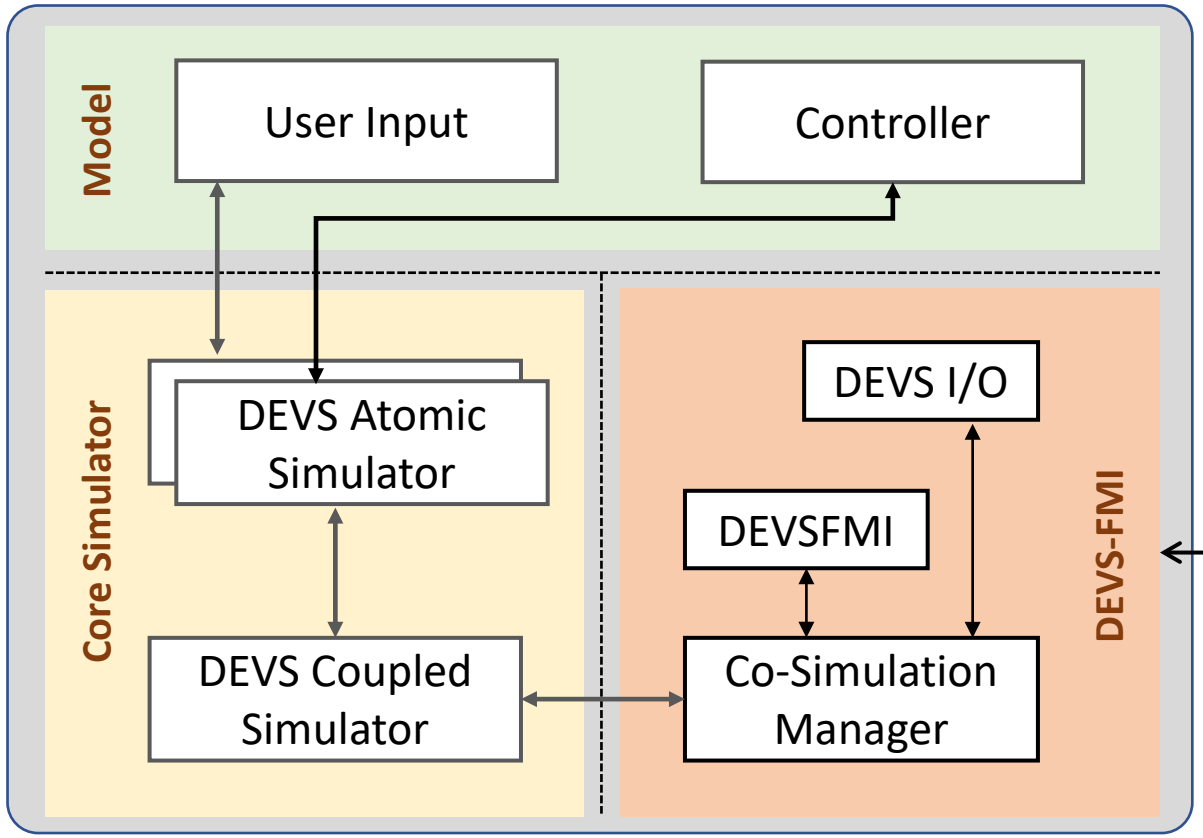


DEVS-Suite simulator

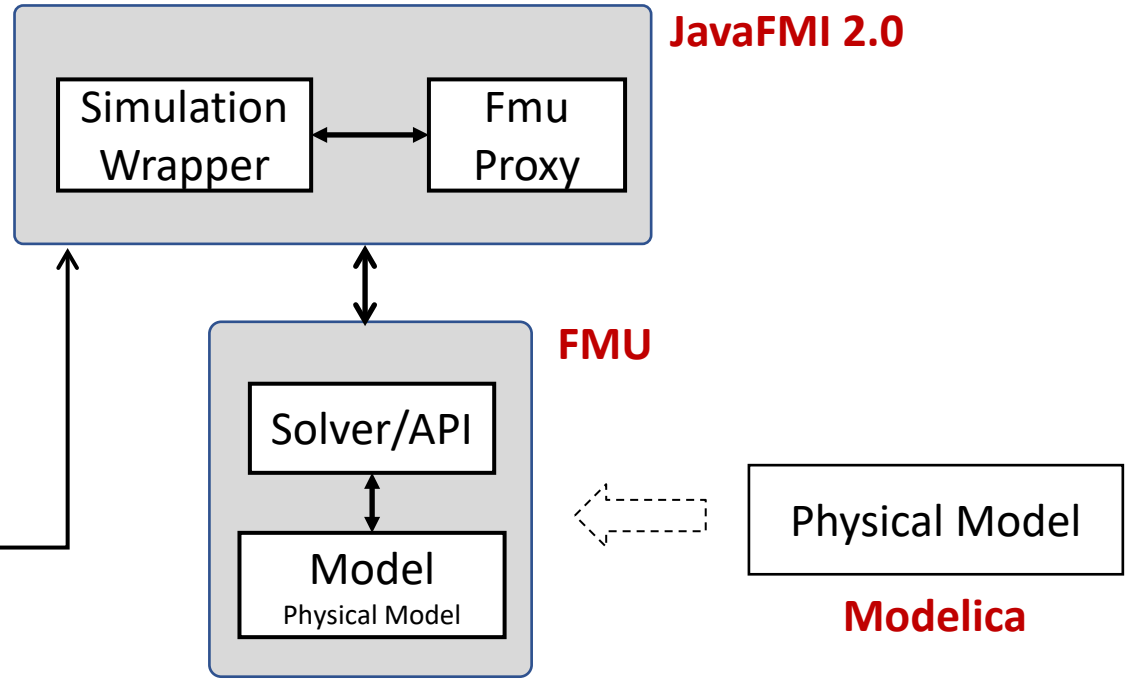
DEVS-FMI Interface

DEVSFMI	generic FMI operations
CoSim Manager	DEVS $\overset{I/O\ mapping}{\longleftrightarrow}$ FMU

co-simulator: DEVS-Suite, DEVS-FMI (JavaFMI)

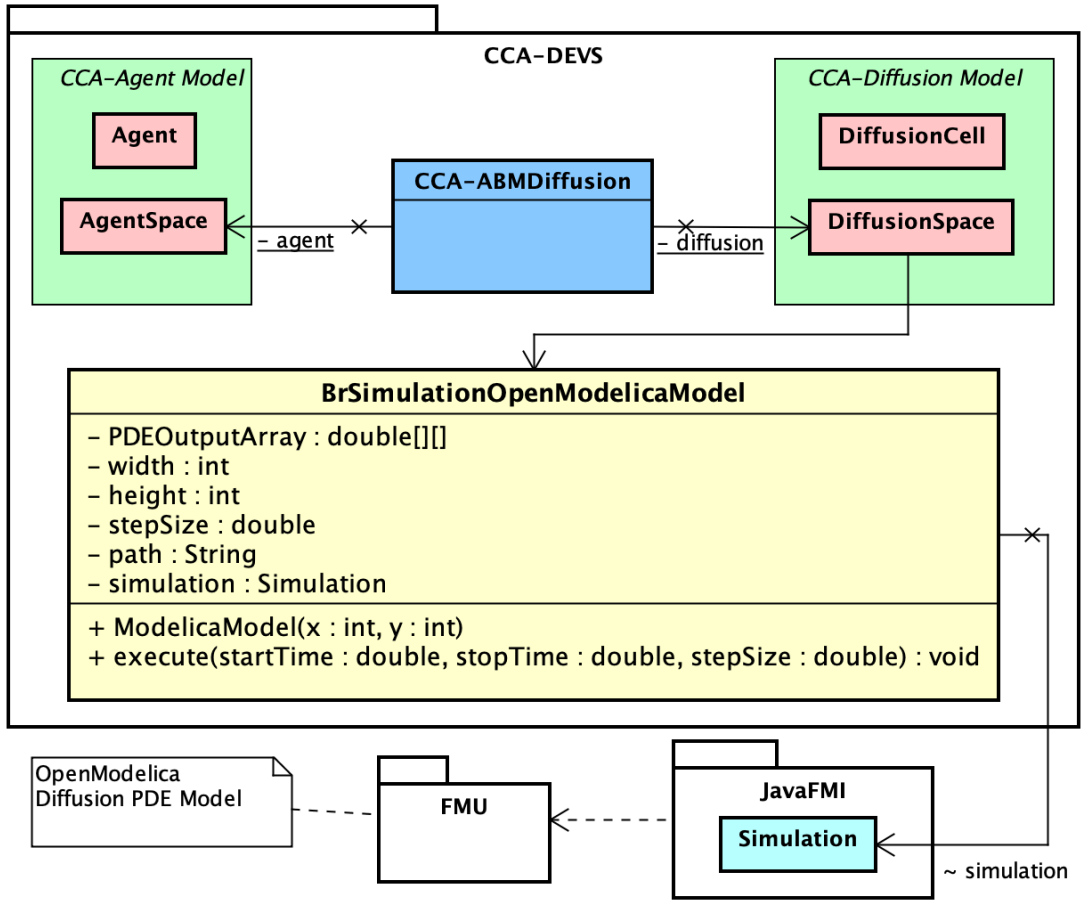


DEVS-Suite



DEVS-FMI Interface	
DEVS I/O	communication & mapping to/from FMU
DEVSFMI	generic FMI operations
CoSim Manager	DEVS $\xleftarrow{I/O\ mapping}$ FMU

co-simulator: DEVS-Suite, JavaFMI



- Composable Cellular Automata are specified in DEVS
- ODE and PDE are specified in Modelica
- Each moving agent communicates with its corresponding diffusion cell. Agents can communicate with their neighbors

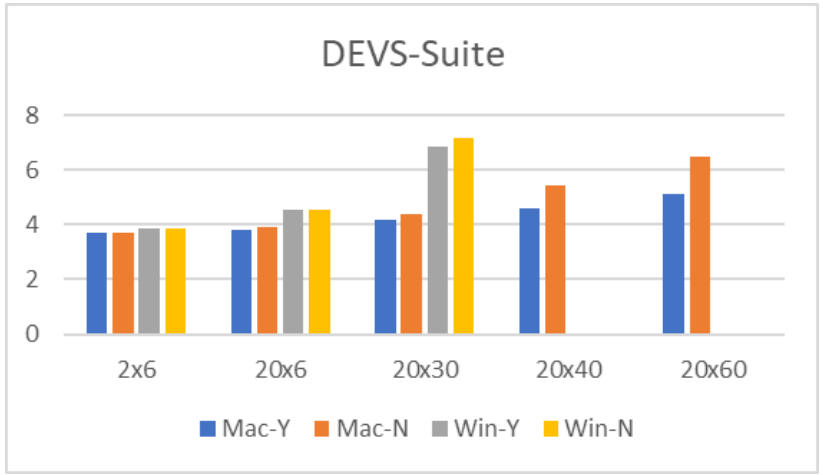
$X \times Y$	6×2	6×20	30×20	60×20
	number of equations			
ODE	228	2,280	11,400	22,800
PDE	24	240	1,200	2,400

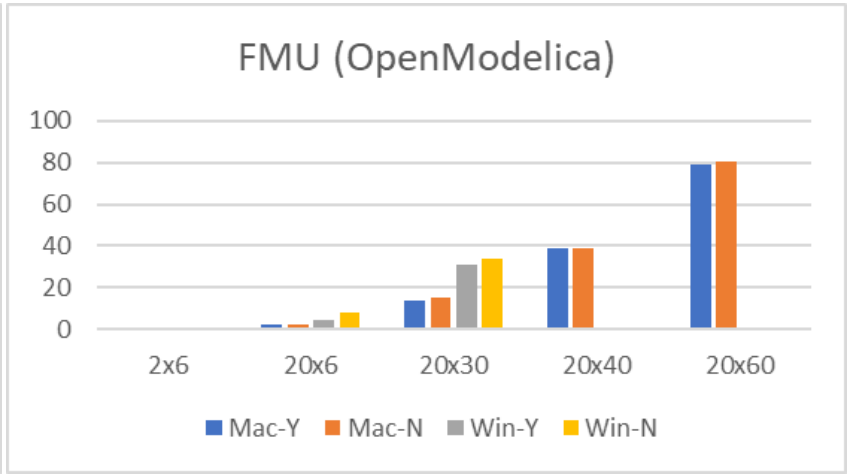
co-simulation computation efficiency (samples)

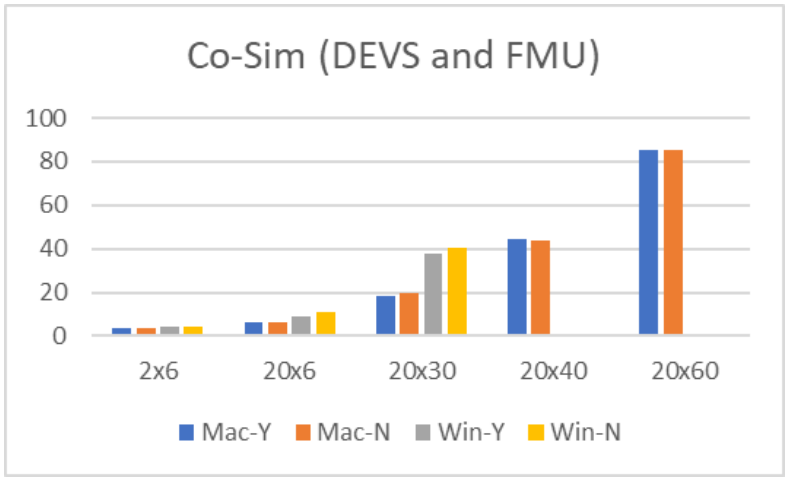
plot X axis: Cellular Automata ($X \times Y$) cells: 12, 120, 600, 800, 1200

plot Y axis: execution time [seconds]

FMU file size linearly scales with CA size







interactions (control & decision)		
	Agent ↔ Agent + Agent ↔ ODE/PDE	**Agent ↔ ODE/PDE**
Mac IOS (64 GB RAM)	Mac-Y	Mac-N
Win 10 (8 GB RAM)	Win-Y	Win-N

conclusions

- Hybrid models developed using formal modeling approaches are key to managing complexity and scale traits of simulating Systems-of-Systems including Cyber-Physical Systems.
- Parallel DEVS and Modelica coupled with FMI form a strong duo for co-simulation of time-sensitive, safety-critical systems.
- DEVS-FMI vs. direct FMI have competing roles for hybrid model modularity/hierarchy vs. co-simulation performance.

Thank You

Q/A

ACIMS
ARIZONA CENTER FOR INTEGRATIVE MODELING AND SIMULATION