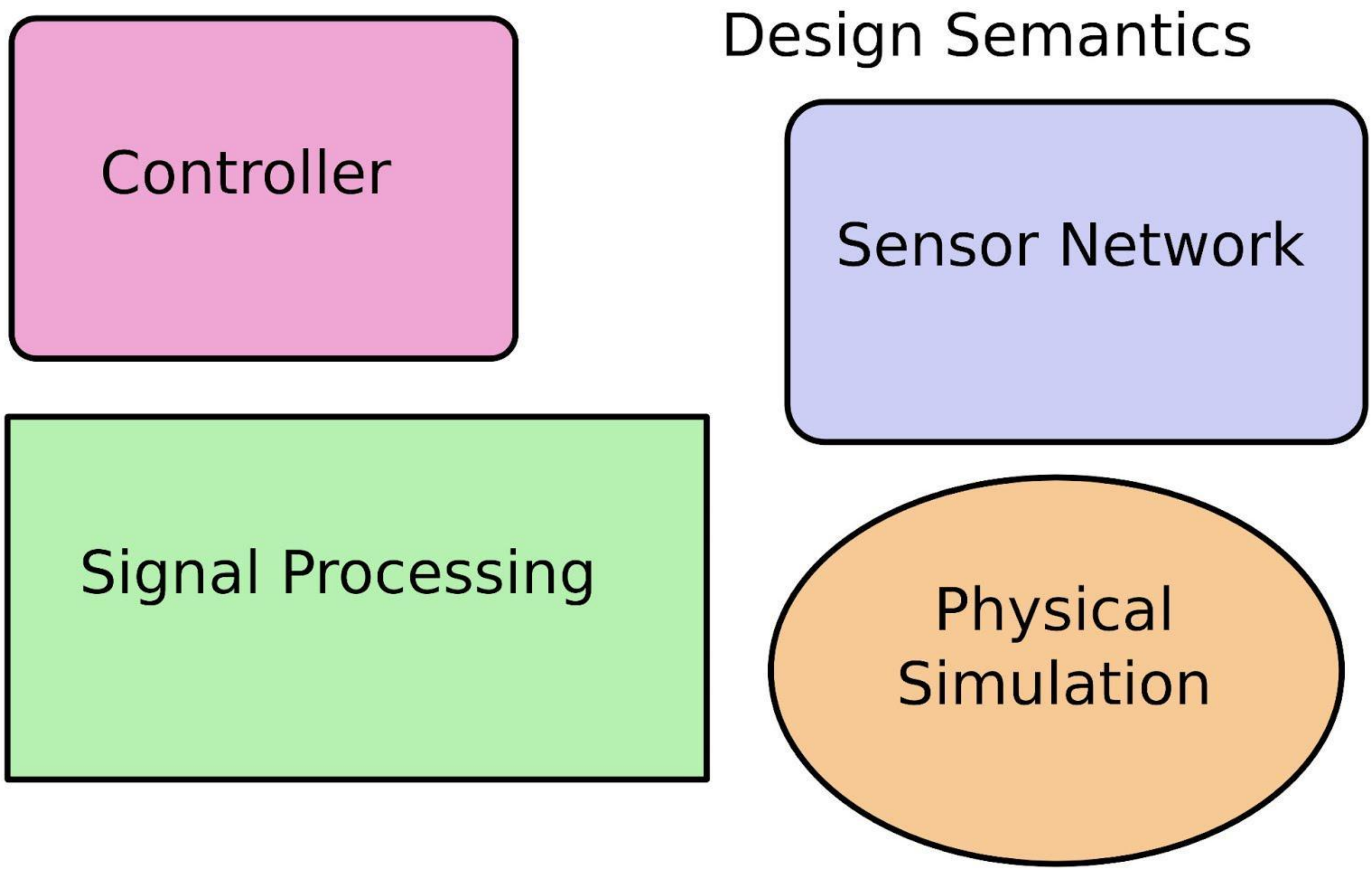
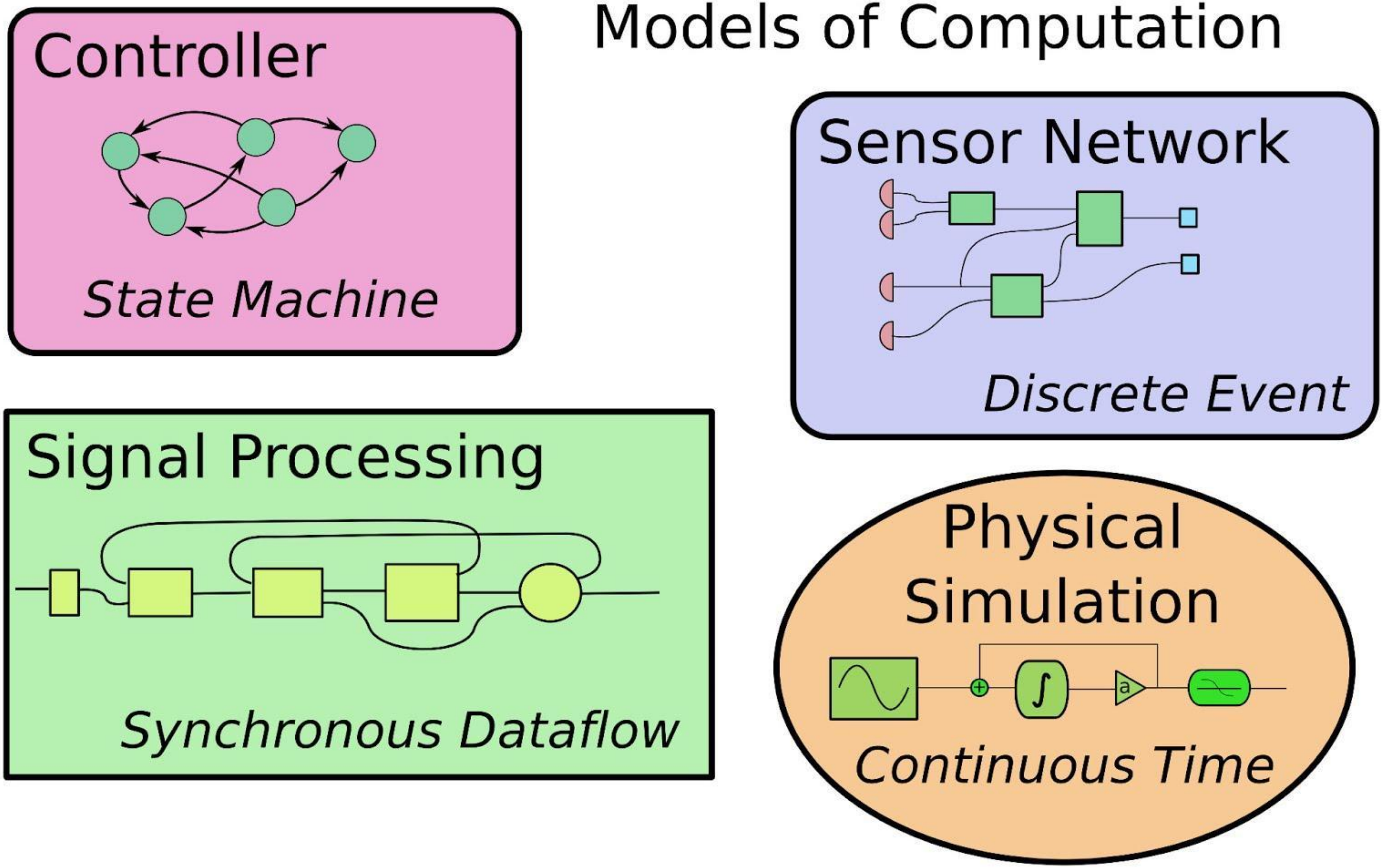
Modeling Heterogeneous Semantics in Ptolemy

with Modular Actor Interfaces

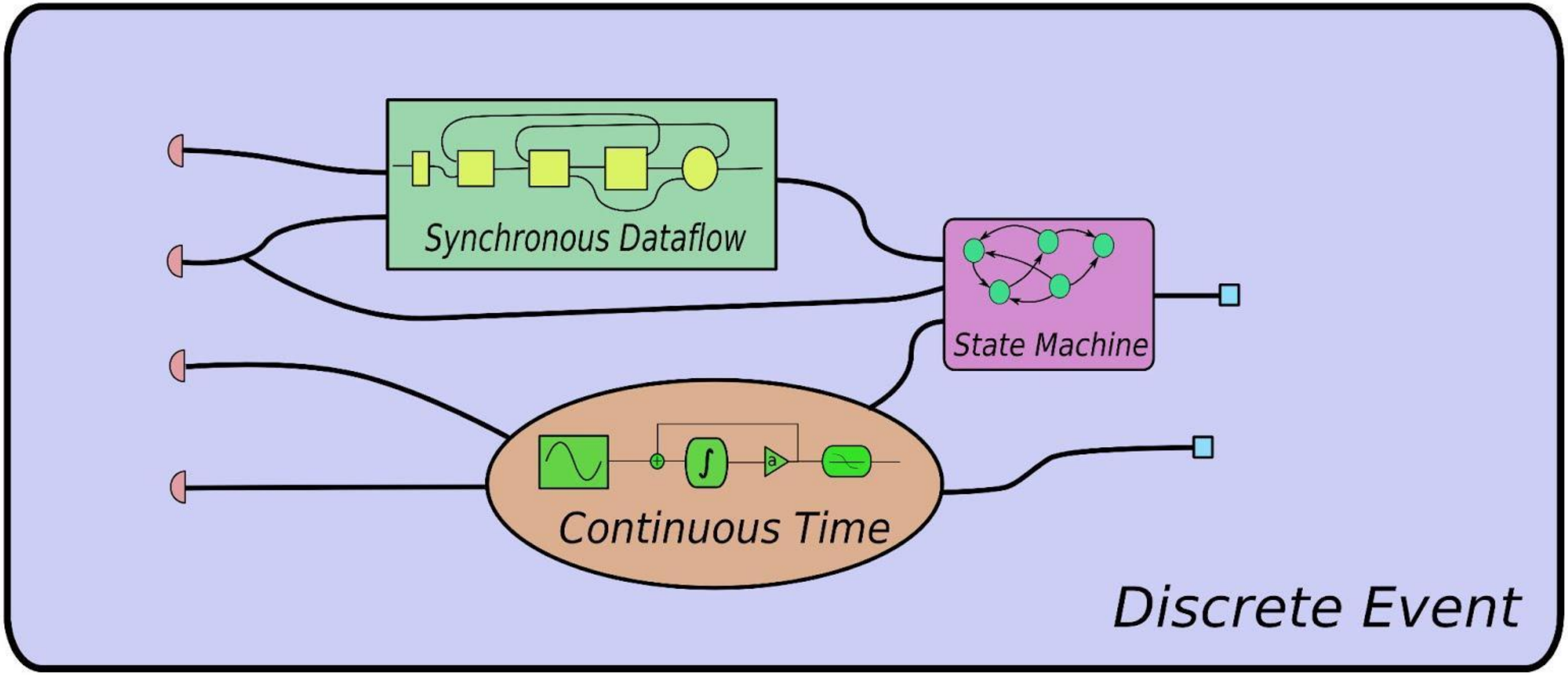
Chris Shaver
working with Professor Edward A. Lee


Design Semantics
Controller
Sensor Network
Signal Processing
Physical
Simulation

Models of Computation

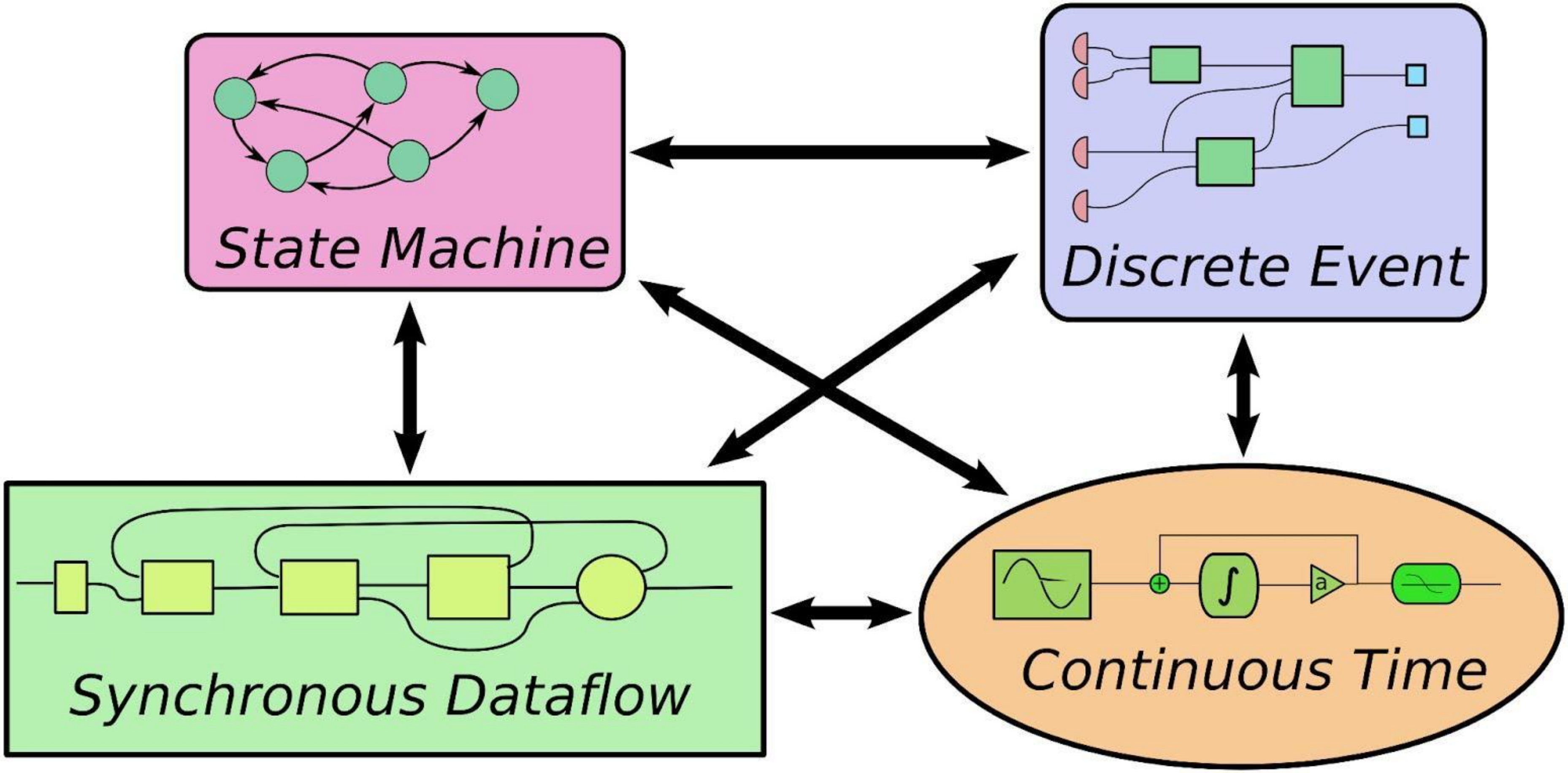


Heterogeneous Systems



Different parts of a system and different levels of hierarchy can have different Models of Computation

Interoperability



Reasoning About Heterogeneity

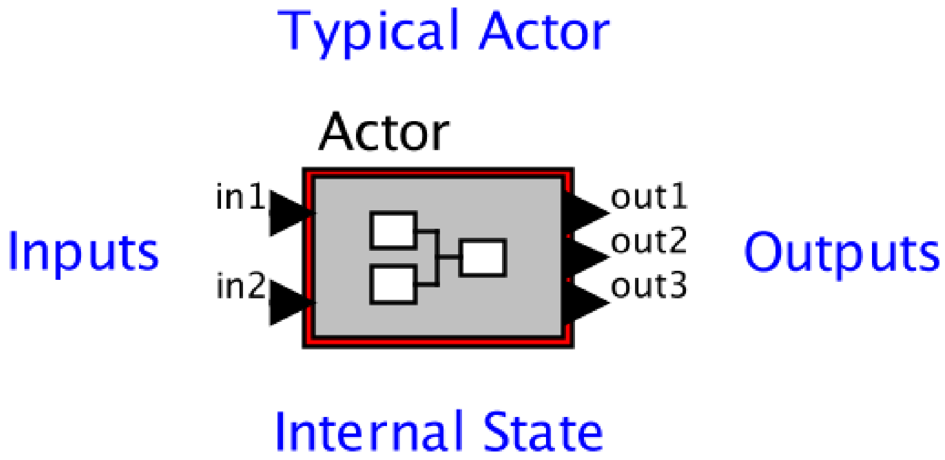
- There needs to be a way to reason about the composition of models with different Models of Computation.

- This is necessary for establishing a consistent semantics for heterogeneous systems.
- This can be used to do analyses and forms of verification that cross the boundaries of heterogeneous composition.

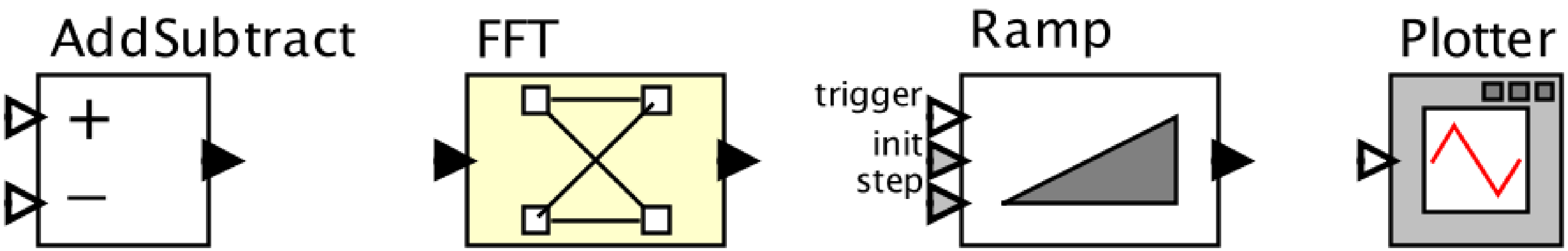
Ptolemy II

- An open source research platform for modeling systems.
- Implements an Actor Oriented language for model development.
- Allows hierarchical design and the heterogeneous composition of different Models of Computation.
- ~2.5 Million Lines of Code.
- Code generation.

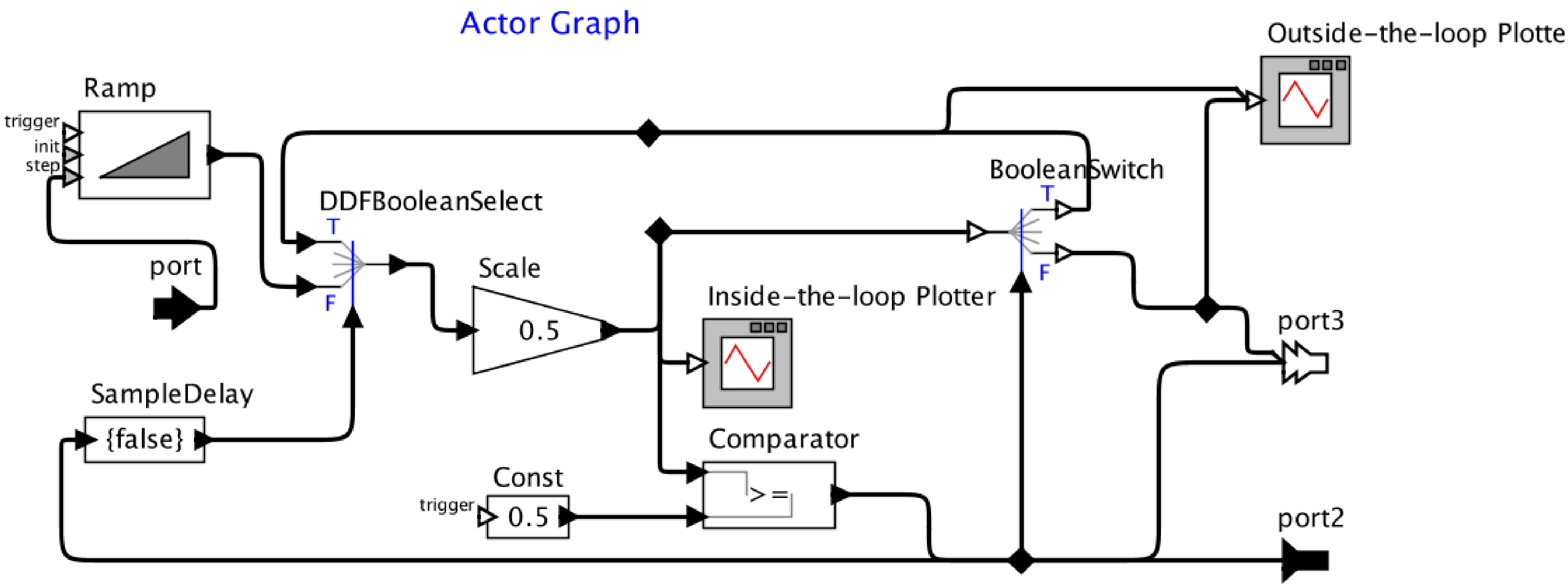
Actors



Some Specific Actors

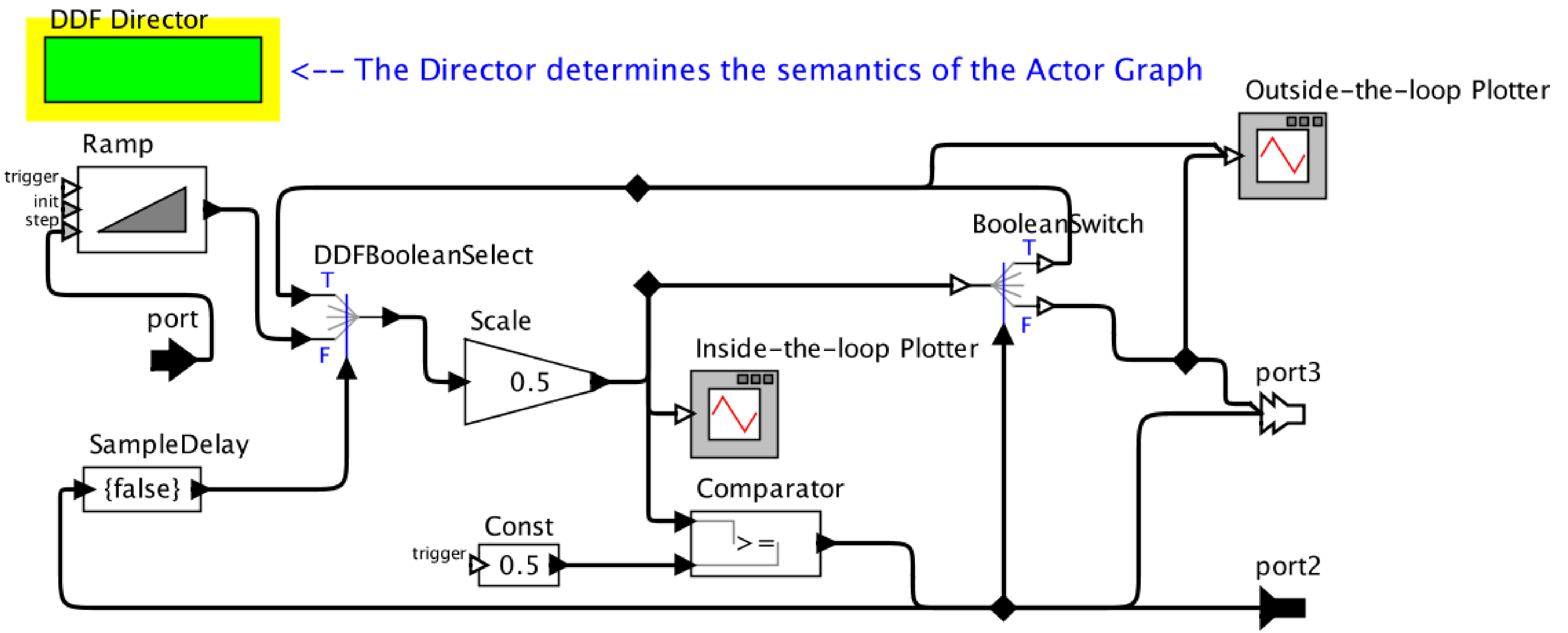


Actor Graphs



Author: Gang Zhou
(based on Ptolemy Classic demo)

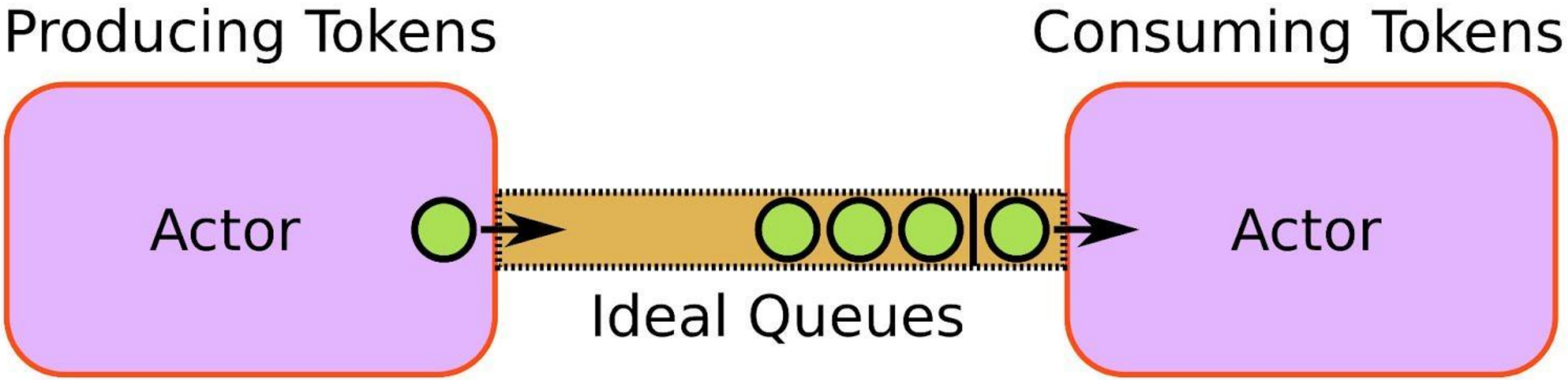
Models



Author: Gang Zhou
(based on Ptolemy Classic demo)

Dataflow Directors :

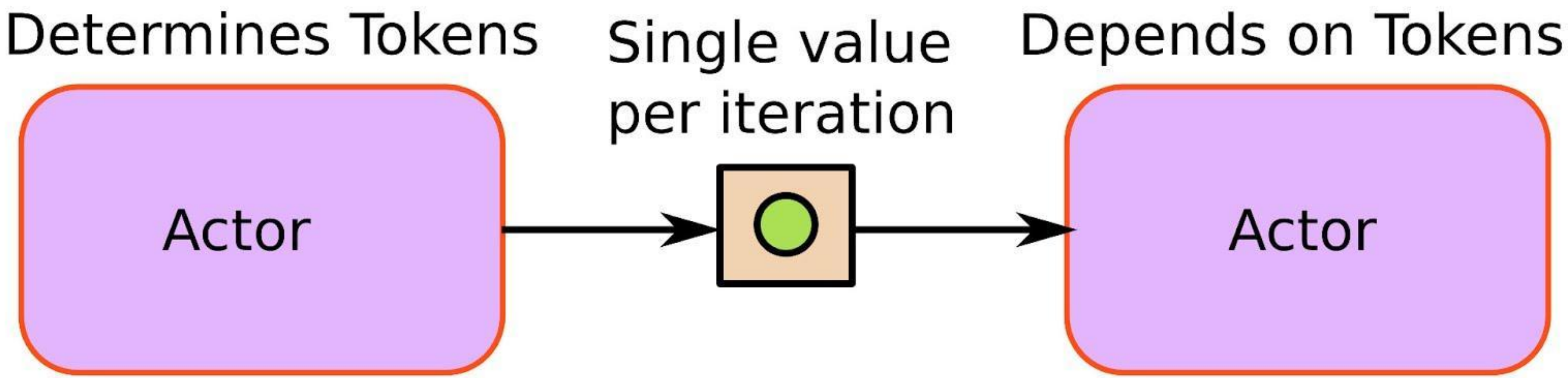
DDF, PSDF, CSDF, MDSDF, HDF, SDF, etc...



SDF : Production and Consumption rates are fixed. Scheduability is decidable, and if a model is scheduable, it can be scheduled periodically with fixed-sized buffers.

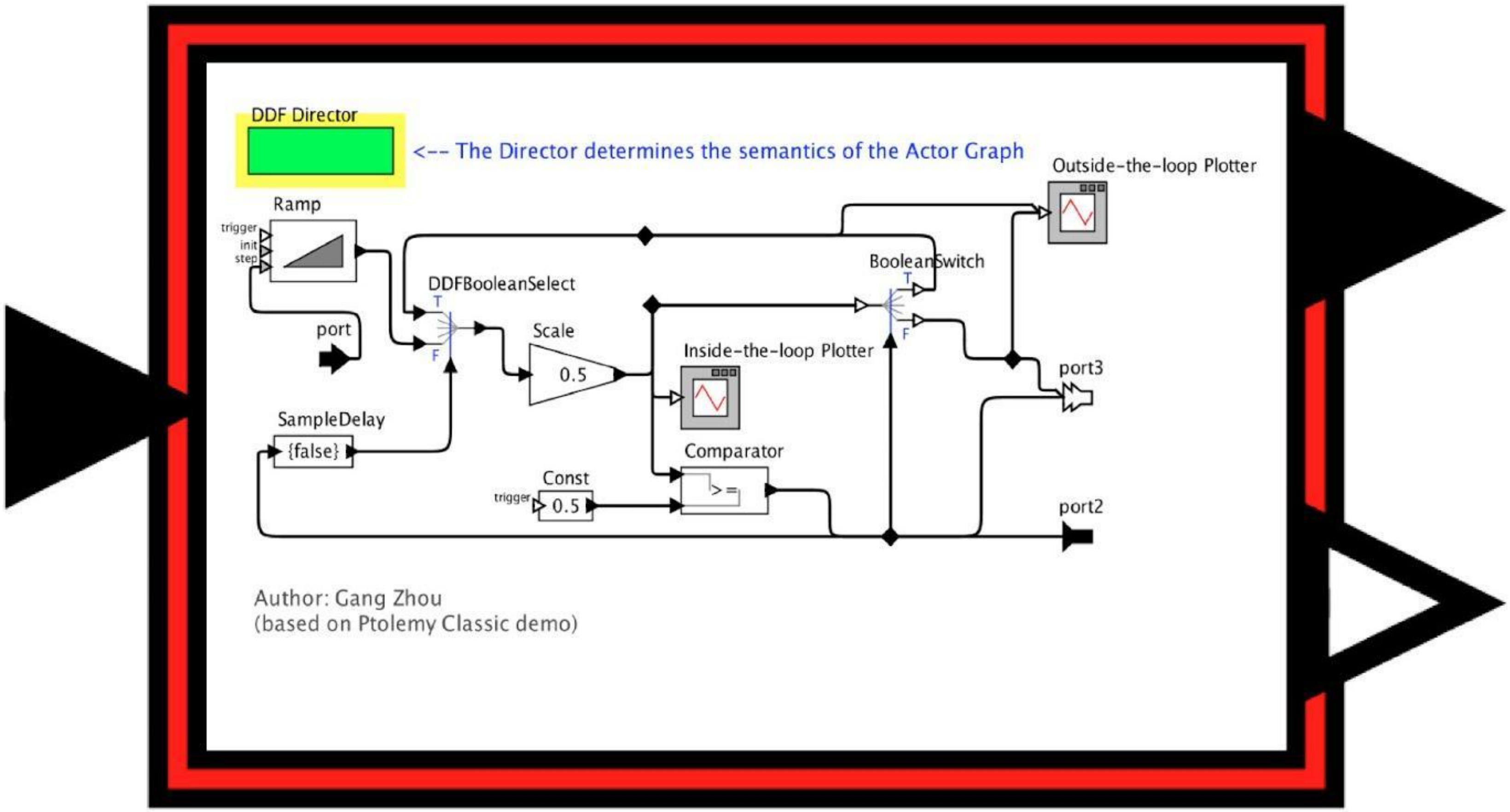
Synchronous Directors :

SR, DE, CT, etc...

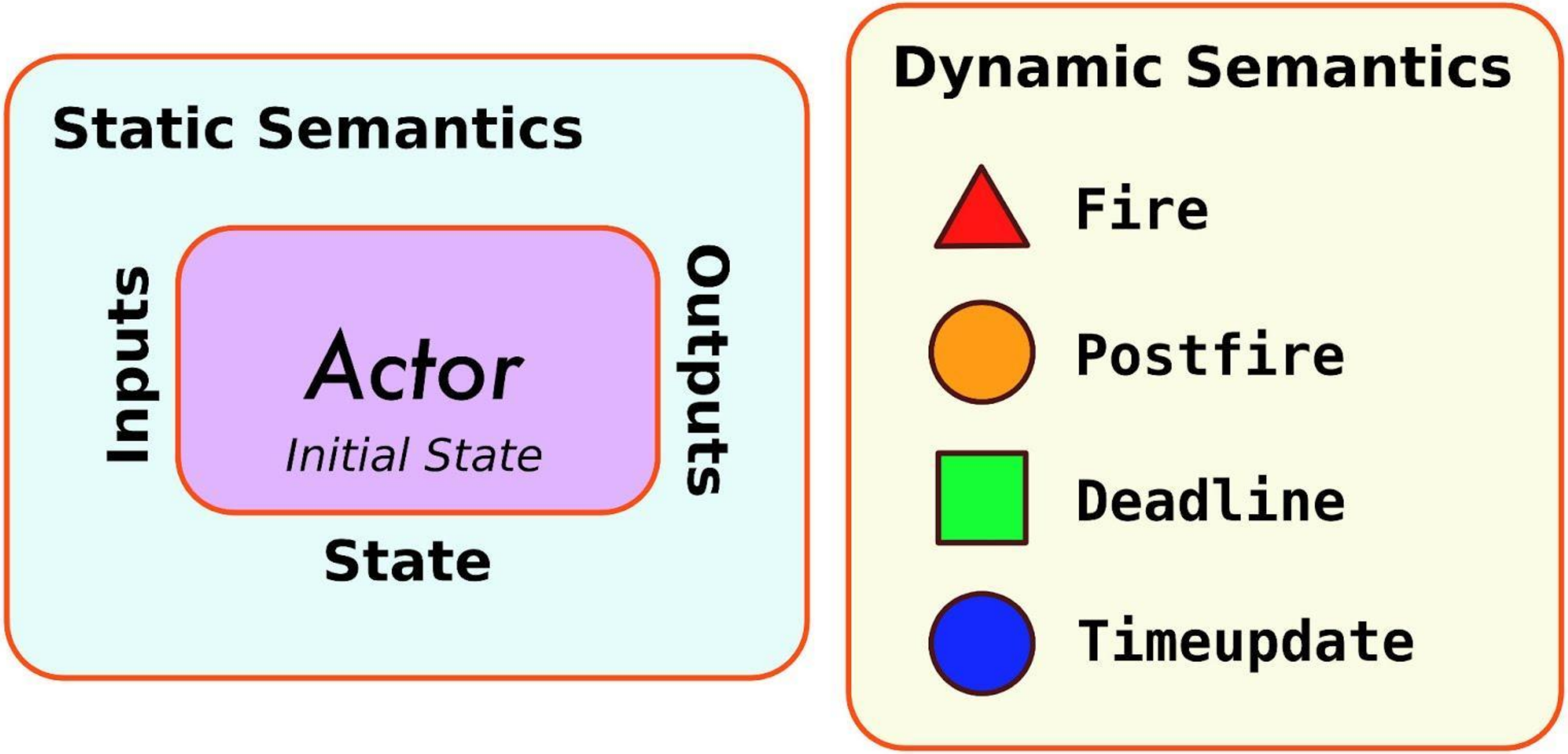


SR : If actors in a model are monotonic then there exists a unique solution for the value of every relation that can be reached through a fixed-point iteration.

Composite Actors

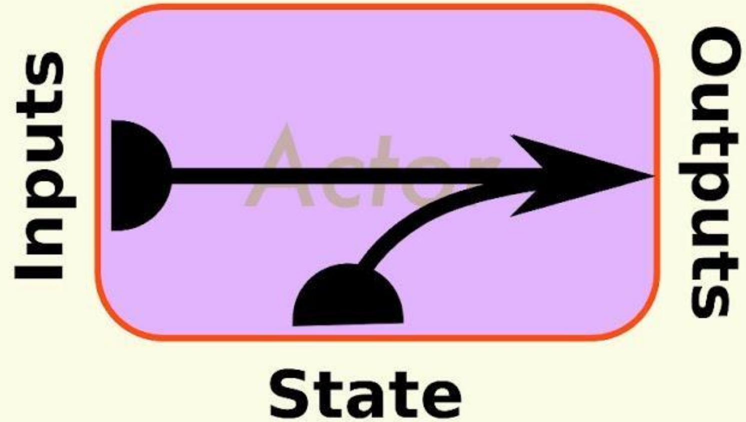


Modular Actor Interface

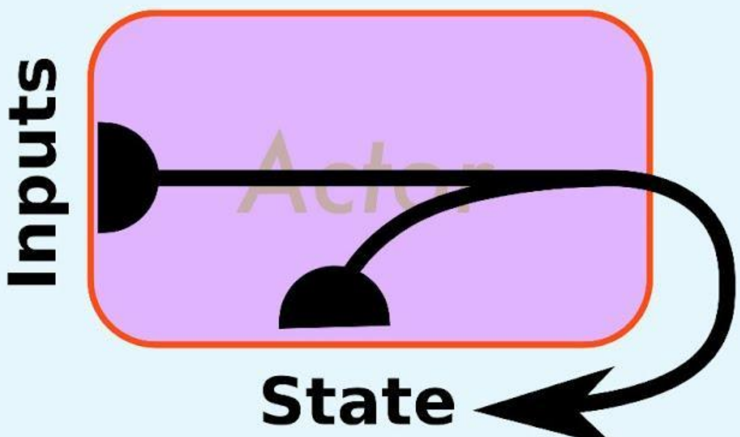


Fire

Produces outputs
from inputs and state



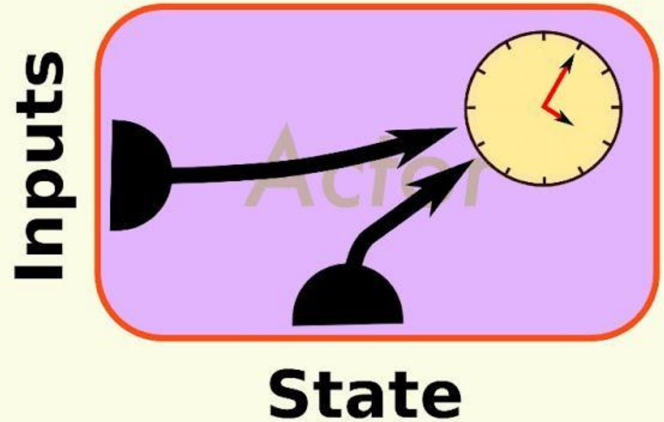
Postfire

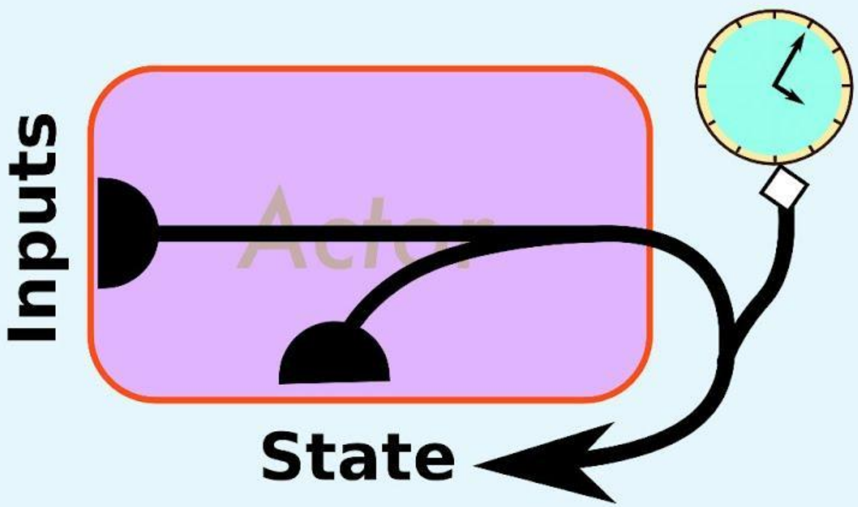


Updates to a new state
from inputs and state

Deadline

Determines a deadline for execution from inputs and state

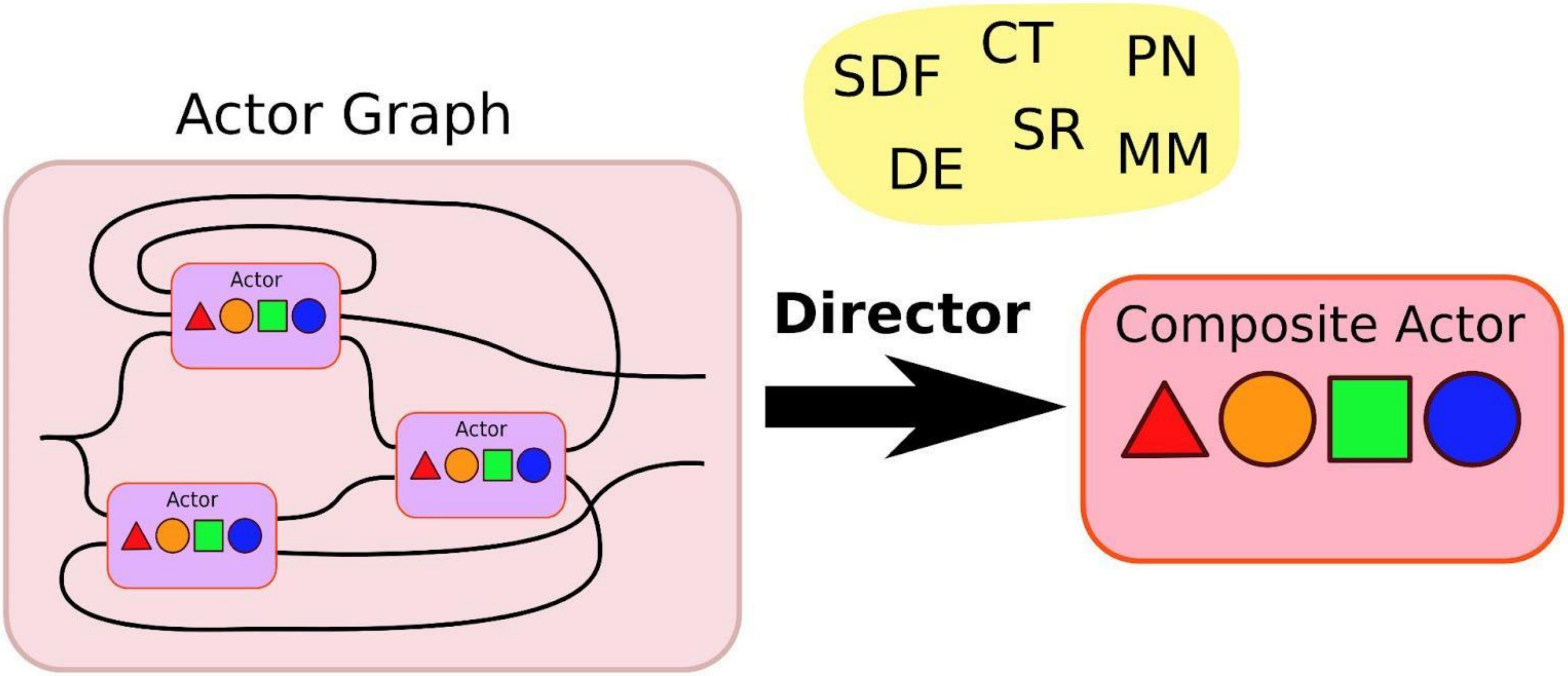




Timeupdate

Updates to a new state from inputs, state, and time

Directors Compose Actor Graphs into Actors



Directors compose the interfaces of Actors in a Model into those of the Composite Actor.

Mathematical Representations

The fixed-point semantics of SR can be understood in clear mathematical terms.

$$\left(\widetilde{F}_{s,x}(y)\right)(o) \quad = \quad F_j\left(s_j, (x, y) \restriction_{I_j}\right)(o)$$

Letting $y^*_{s,x}$ be the unique least fixpoint of $\widetilde{F}_{s,x}$.

$$\begin{aligned} F(s,x) &= y^*_{s,x} \restriction_O \\ P(s,x) &= \left(P_1\left(s_1, (x, y^*_{s,x}) \restriction_{I_1}\right), \ldots, P_n\left(s_n, (x, y^*_{s,x}) \restriction_{I_n}\right)\right) \end{aligned}$$

From "*A Modular Formal Semantics for Ptolemy*", Tripakis et al.

Coroutine Model of Computation

Theorem 1. *Given a non-strict Coroutine Model $\mathcal{M}$, if the input $\mathbb{I}_{\mathcal{M}}$ and output $\mathbb{O}_{\mathcal{M}}$ types of the model are finite-height pCPOs and operator $\oplus$ is monotonic, then the above recursive equations characterizing the kernel functions* $\mathbf{e}$ *and* $\mathbf{f}$ *have unique least fixed-point solutions in the partial order of functions with codomains $2^{\mathbb{G}}$ and $\mathbb{O}_M$, respectively.*

Theorem 2. *Given a non-strict Coroutine Model $\mathcal{M}$, if the input $\mathbb{I}_{\mathcal{M}}$ and output $\mathbb{O}_{\mathcal{M}}$ types of the model are finite-height pCPOs and operator $\oplus$ is monotonic, and if for each $q \in Q$ the functions $\mathbf{enter}_q$ and $\mathbf{fire}_q$ are monotonic in terms of $\mathbb{I}_q$, and the mapping functions $m_{\mathbb{I}}$ and $m_{\mathbb{O}}$ are monotonic, then the non-strict kernels* $\mathbf{e}$ *and* $\mathbf{f}$ *are continuous in terms of $\mathbb{I}_{\mathcal{M}}$.*

"*The Coroutine Model of Computation*", Shaver and Lee, 2012

I Conclude !

- Try out Ptolemy! It's open and free!
- Check out Modular Actor Interfaces.
- Feel free to design your systems modularly, hierarchically, and heterogeneously.
- http://ptolemy.eecs.berkeley.edu