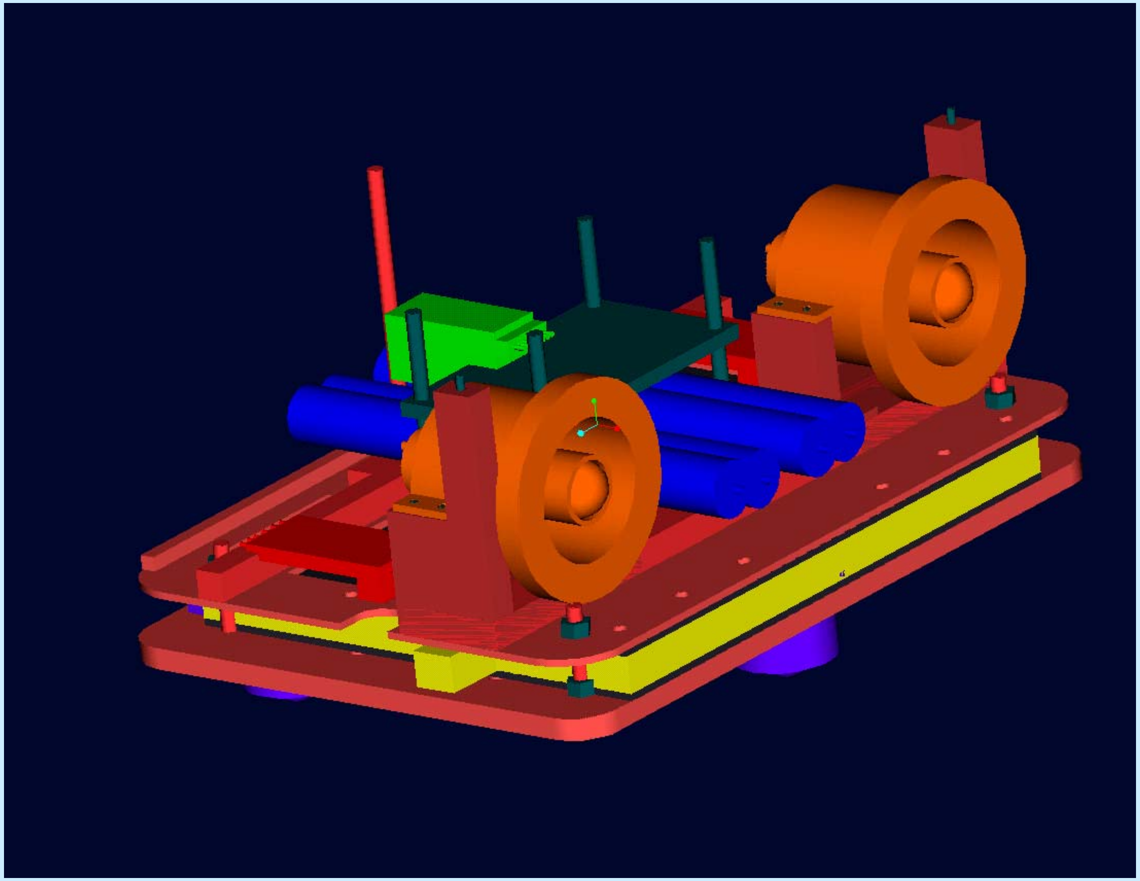
Modeling and controlling the Caltech Ducted Fan Vehicle



Steve Neuendorffer, Ptolemy Group, UC Berkeley

How do we get to an implementation?

Abstract, high-level system model

Detailed low-level system implementation

Some options…

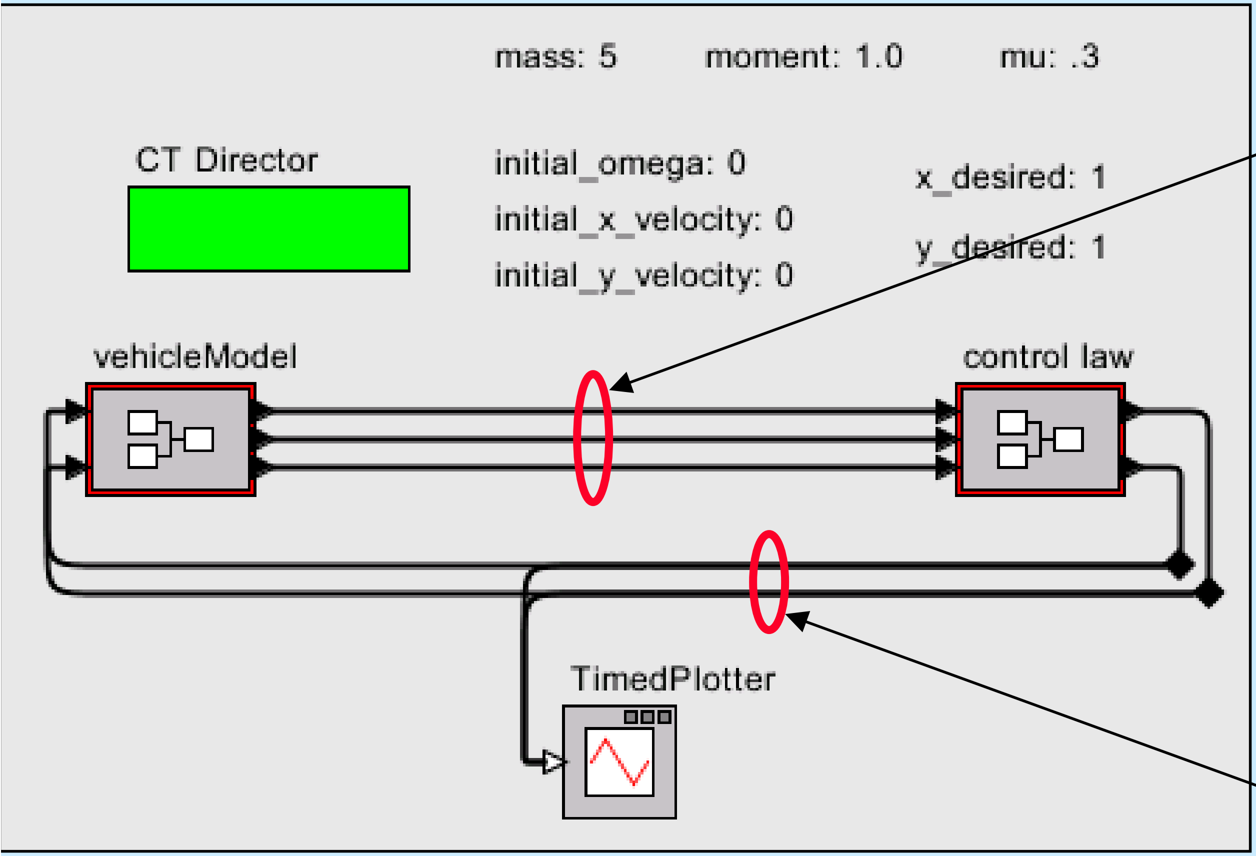
- Don't build a high-level model… Implement the system by hand.
- Build a high-level model, but throw it away when implementation starts.
- Build a high-level model, and validate implementation against it.
- Build a low-level model, and generate an implementation from it.
- Build a high-level model, refine it gradually into more detailed models that are more suitable for automatic code generation.

An example of heterogenous refinement

Drive the ducted fan vehicle to a desired point.

- Continuous time vehicle and controller.
- A zero-delay discrete-time controller.
- A one-step delay discrete-time controller.
- An arbitrary delay discrete-time controller.
- A more sophisticated modal controller.
- A model for automatic system implementation.

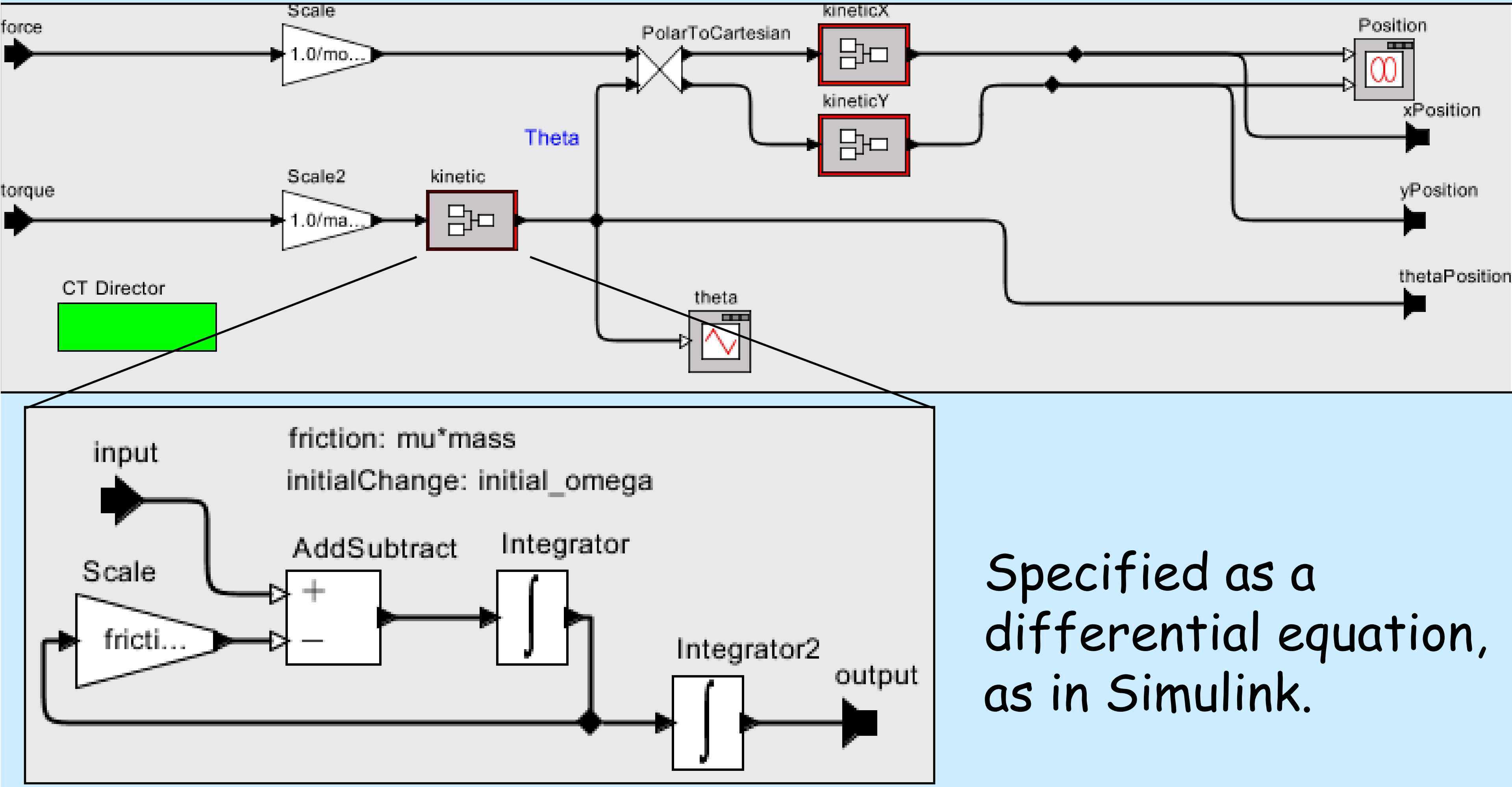
1: Vehicle/Controller model



(X,Y) Position, and direction of the vehicle

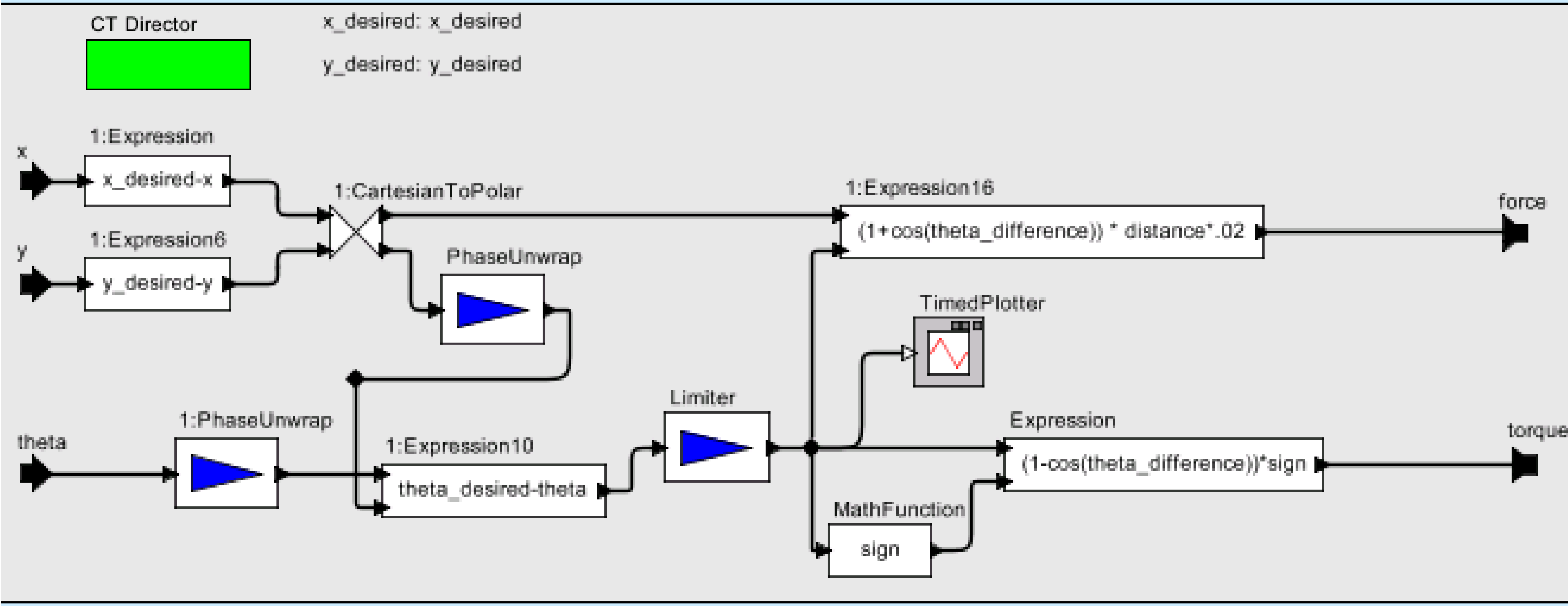
Total forward thrust, and differential torque applied by the fans.

1: Continuous Vehicle model



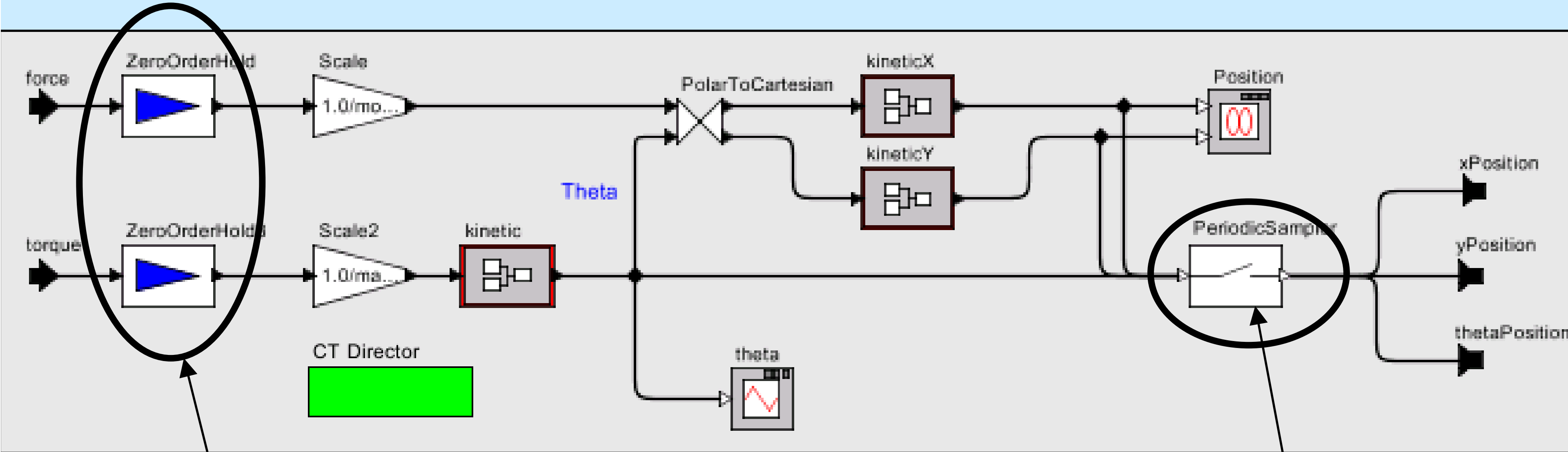
Specified as a differential equation, as in Simulink.

1: Continuous controller model



A modified proportional controller…

2: Vehicle model with Discrete-time interface

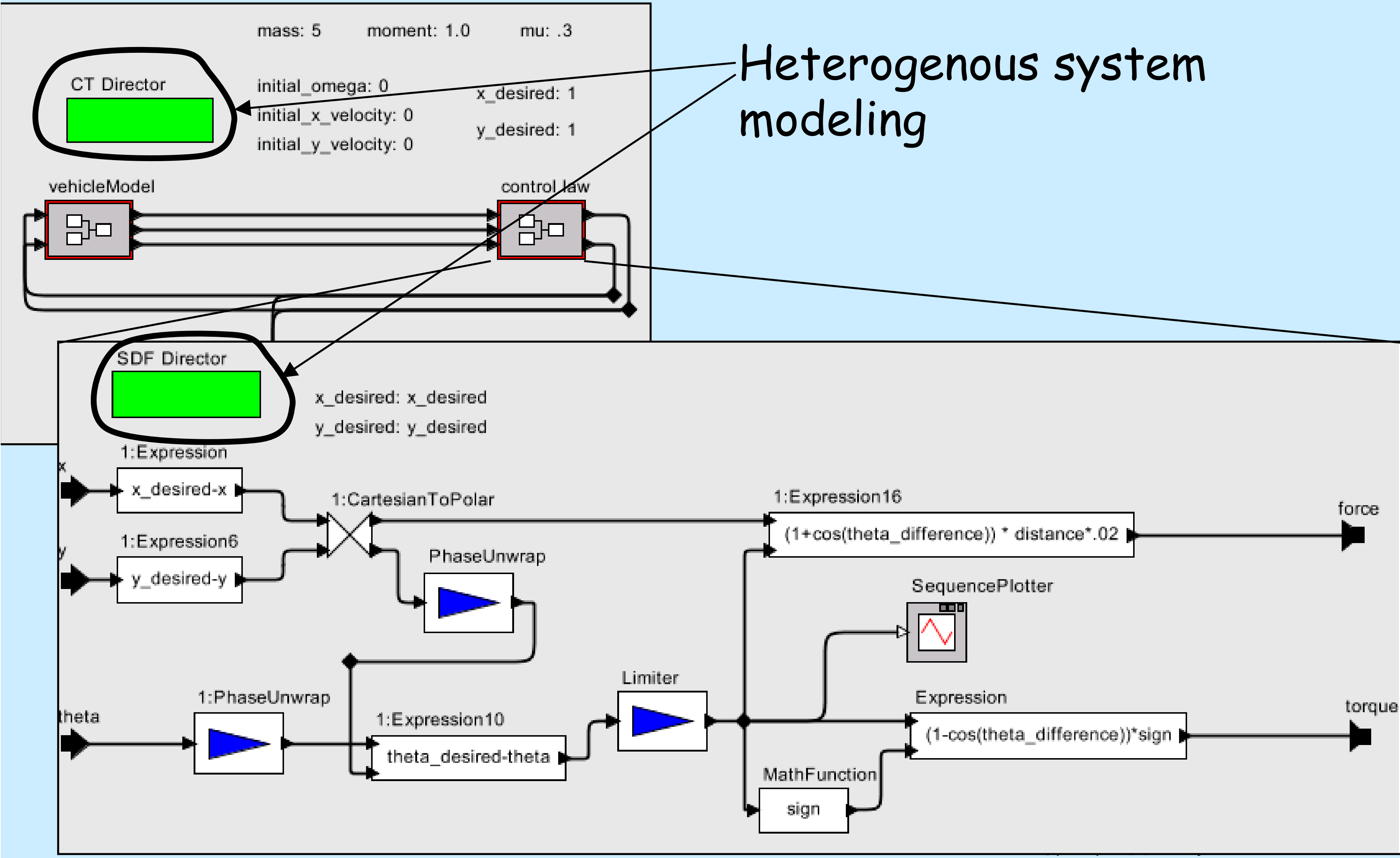


Zero-order hold models the Digital -> Analog conversion.

The analog position of the vehicle is periodically sampled approximately every second.

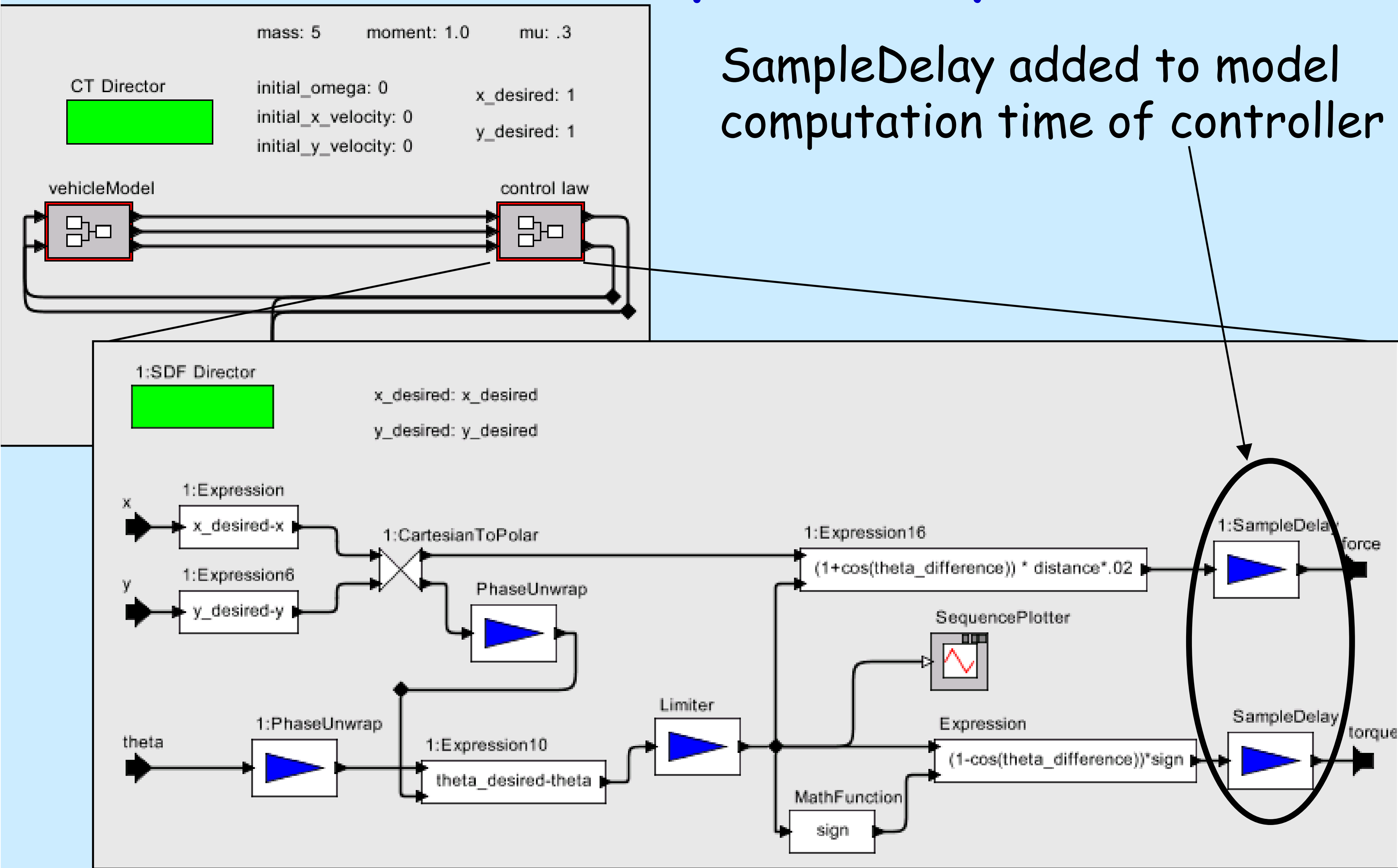
2: Discrete Vehicle controller

Heterogenous system modeling



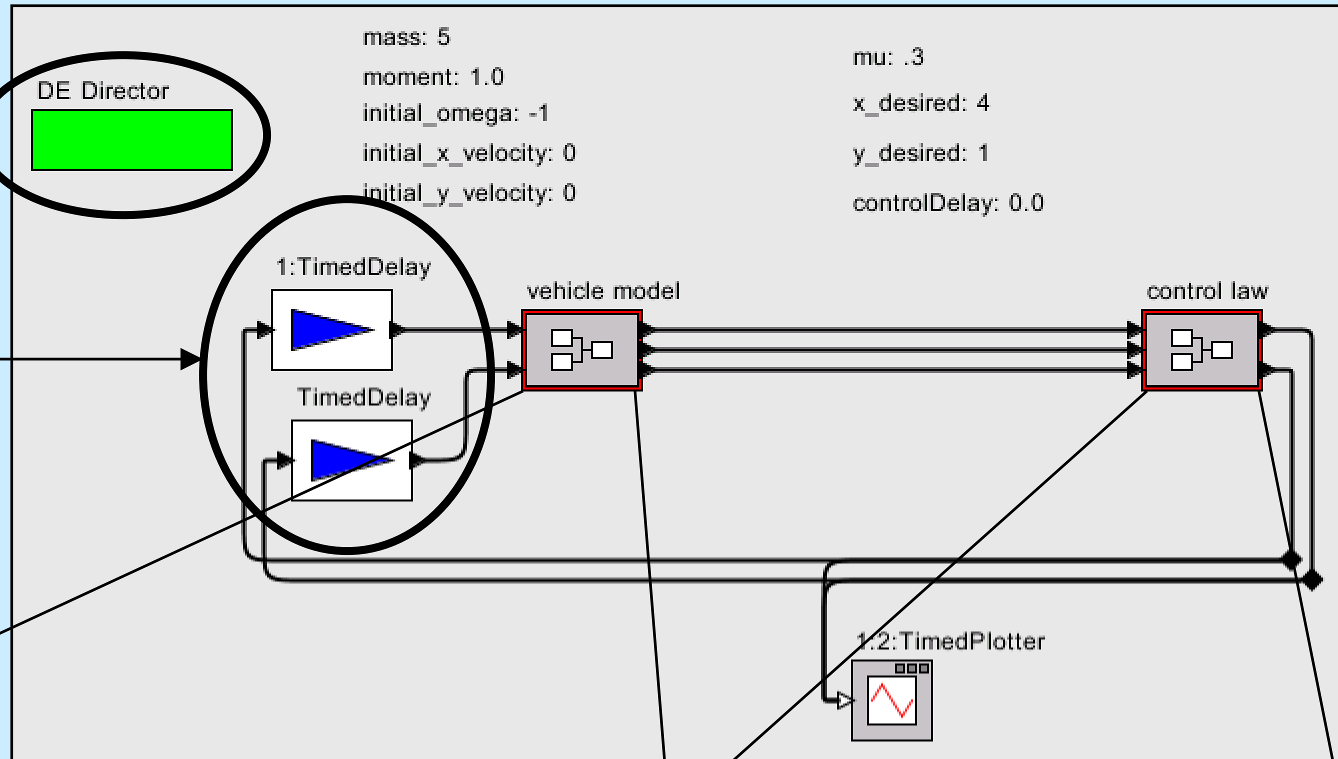
3: Discrete Vehicle controller with one-sample delay

SampleDelay added to model computation time of controller

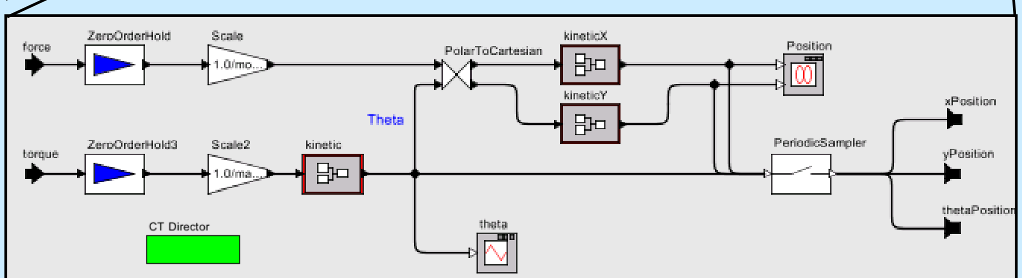


4: Discrete Vehicle controller with arbitrary delay

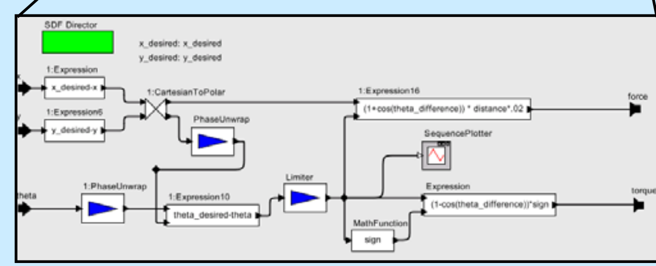
TimedDelay actor in Discrete Event domain models arbitrary delay.



More Heterogeneity

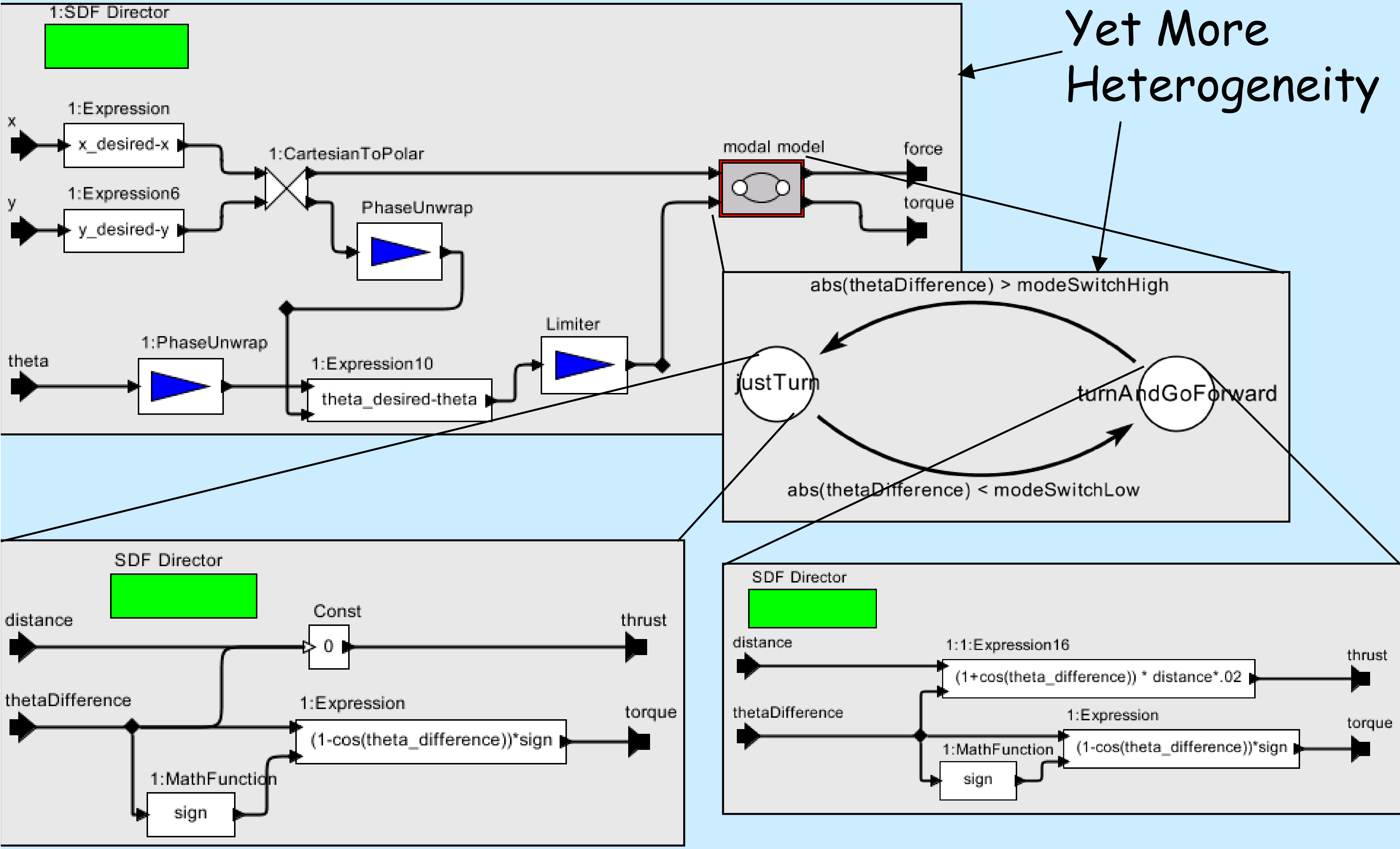


CT vehicle model



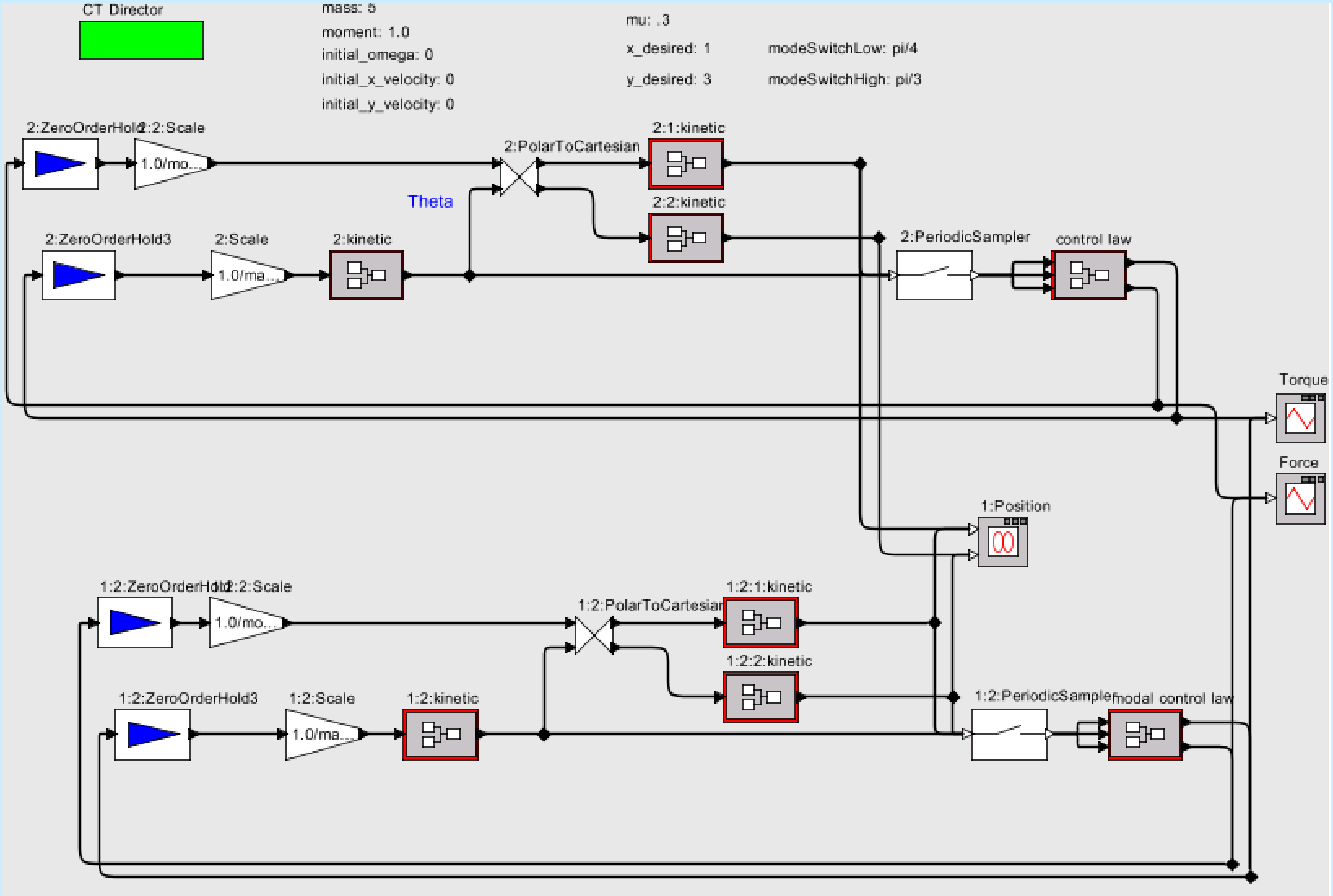
SDF control law

5: A modal controller

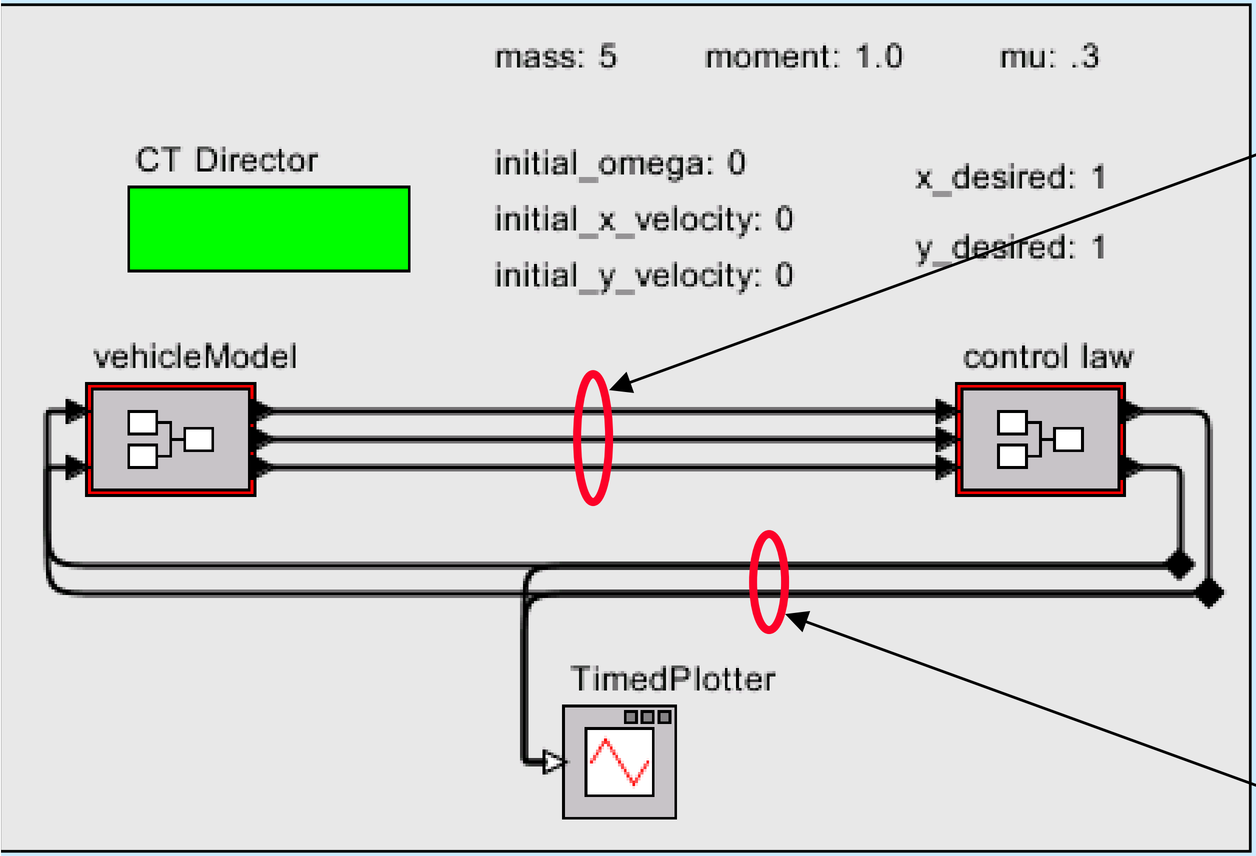


Breathe and watch the demo…

A comparison of the modal controller versus the simple proportional controller.



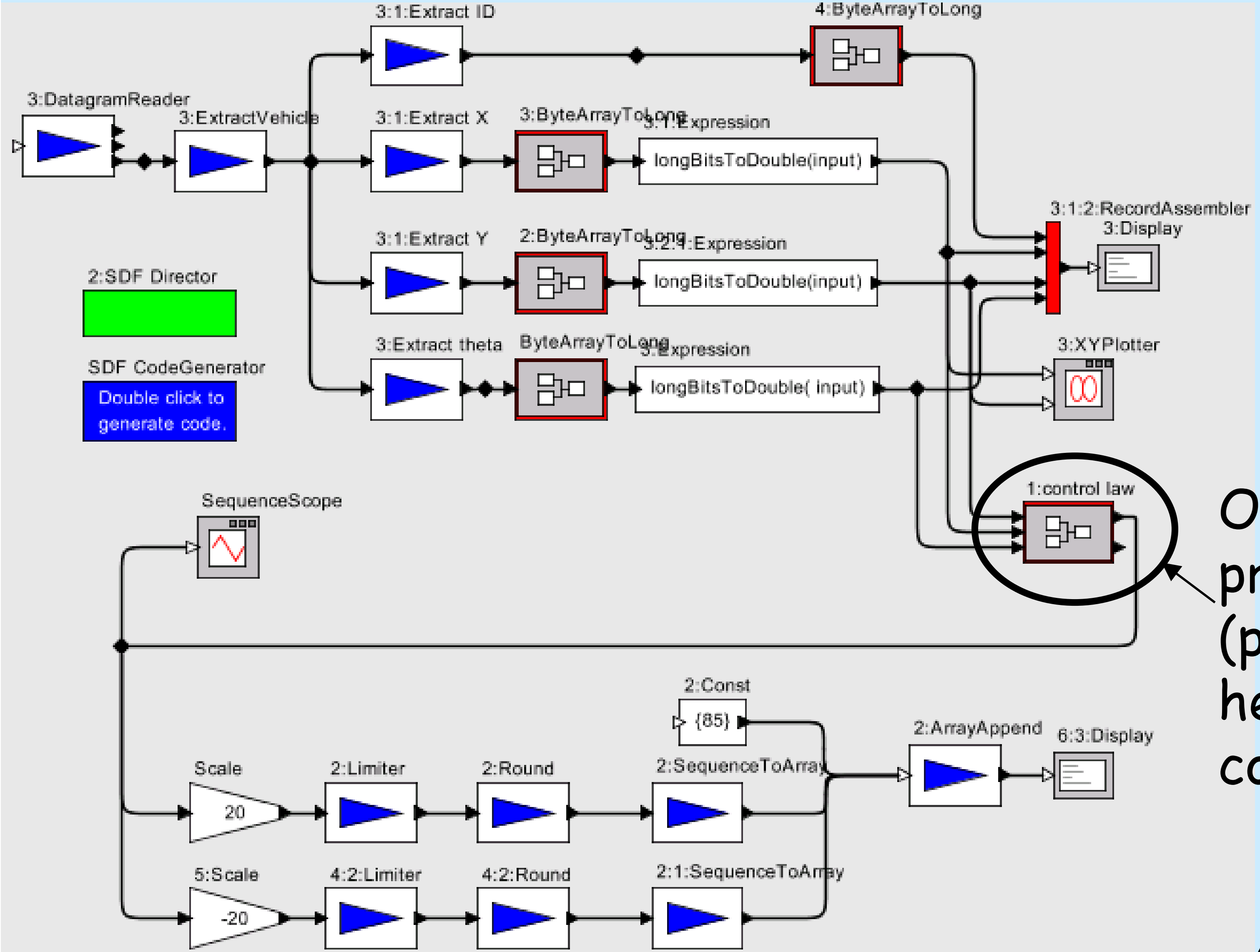
Target specifics…



(X,Y) Position, and direction of the vehicle come from video localization system.

Control values are sent to motor controller by serial port.

6: A controller, with refined communication.



Suitable for code generation

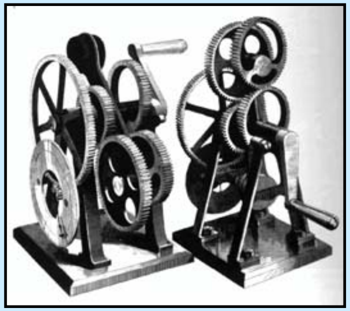
One of the previous (possibly heterogenous) control laws

Code Generation

Translation from a model to an efficient implementation is a highly skilled operation that is both error-prone and time-consuming.

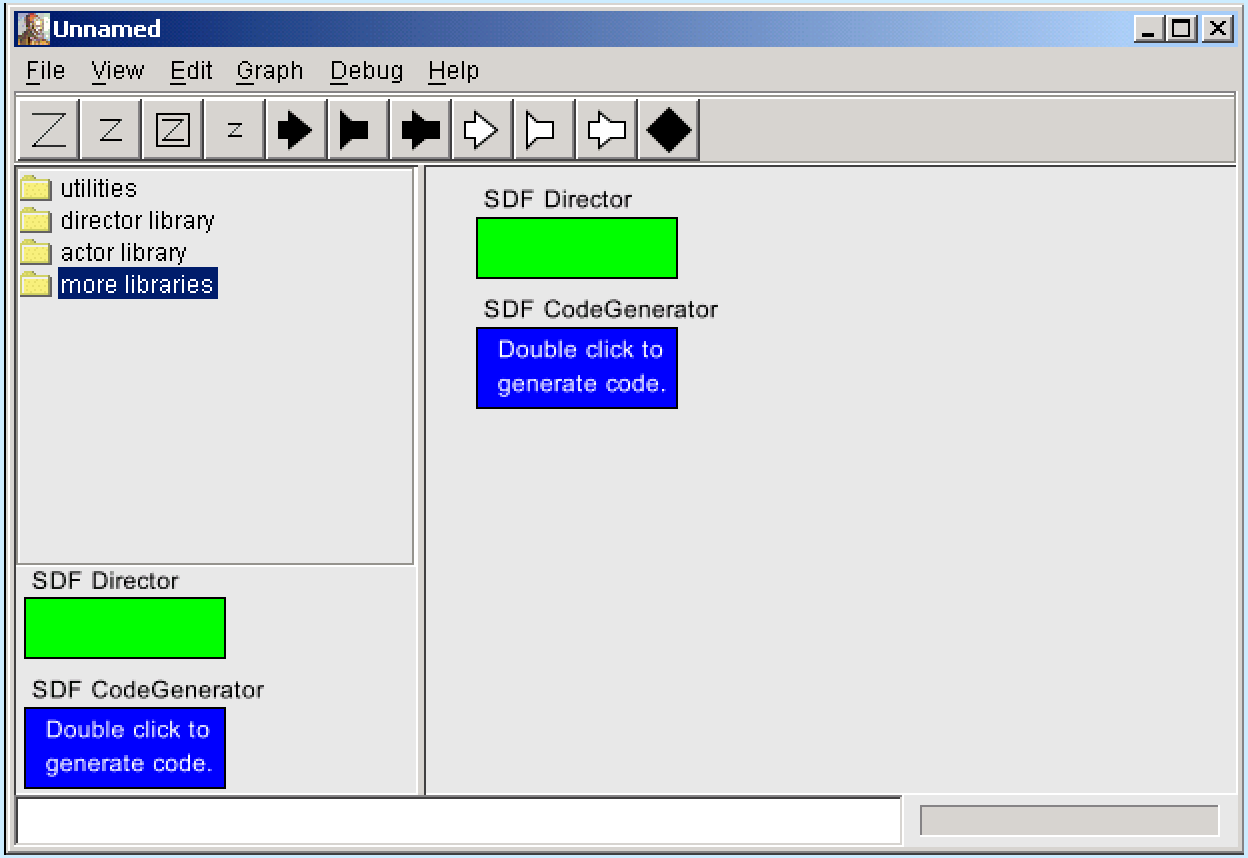
But, it is mostly a repeated application of well-known implementation patterns and optimization techniques.

These patterns and techniques can be automated into modeling tools.



Implementation team = System Modeler + Modeling Tool

Implementation Patterns in Ptolemy II



Task 2.2: Customizing frameworks with generators
Task 2.4: Generating embedded software from models

Code generation

There are two parts to the generated code:

- Code generated from the model of computation
- Code generated from individual components.

How is the code from individual actors generated?

Option 1: The code generator can be implemented specifically for each component.

Option 2: The code generator can be implemented generically for all components.

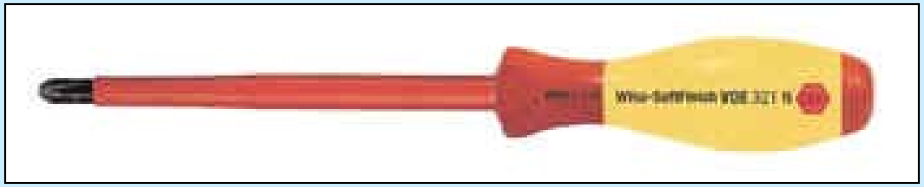
However, being generic and efficient is difficult.

Mapping Ptolemy II Actors

Actors and embedded code have very different design contraints.

Actors are:
- parameterizable
- reusable
- reconfigurable
- generic

Embedded code is:
- specialized
- optimized



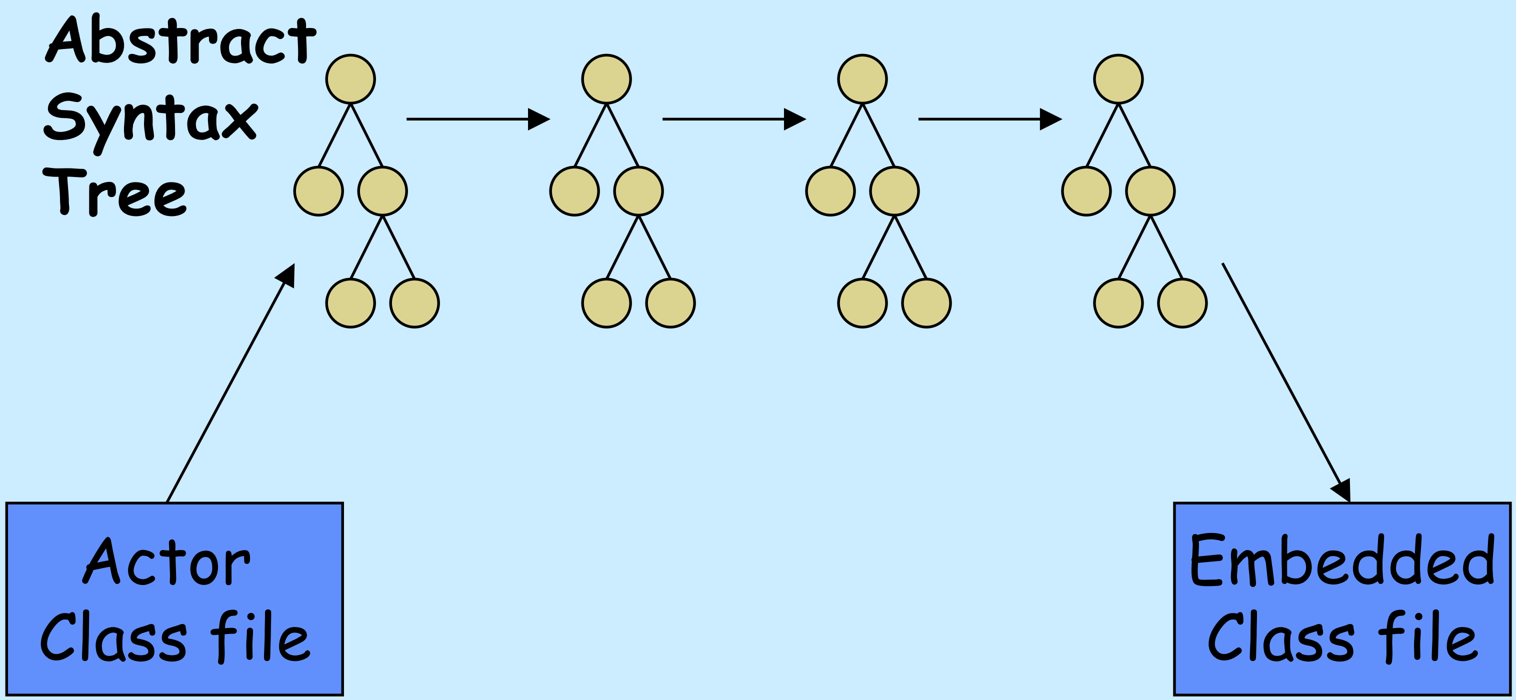
These are inevitably in conflict!

Mapping Java Actors

We must avoid having a multitude of expensive, specialized implementations for each target platform.

Method: Parse the Java code for an actor and specialize it.

Co-compilation



Two main operations:

- Optimize in the context in the model. (Inline parameter values)
- Replace ports with communication primitives of platform.