Adding a Ptolemy Domain...

- Can create new design capabilities while leveraging the Ptolemy architecture and design resources.

- Requires thorough understanding of your computational specialization.
  - refer to existing Domains for examples.

Remember...

- Don't reinvent the wheel.

- It's not commercial software... some adventure and risk.
Typical Domain Extensions (cont.)

- **NEWDomain** - Supports creation of inter-domain mechanisms like NEWToUniversal and NEWFromUniversal, and NEWWormhole.

- **NEWWormhole** - Dual-natured building block at domain interfaces.

- **NEWTarget** - Supervises execution within target system; used by code generation domains mostly.
Since Ptolemy is OO, many extensions, specializations are possible; some are required to have an operational domain.

- **NEWScheduler** - Sequences execution of blocks based on domain semantics.

- **NEWPortHole** - Supports data transfer mechanism for the domain.

- **NEWStar** - Enforces sequence of getting/sending data or events relative to execution. Specific functional stars inherit this behavior.

- **NEWTo/NEWFromUniversal** - Builds correct PortHole/EventHorizon combos for NEWWormholes; implements Wormhole data transfer.
Mechanics: Mkdom and Domain Class Templates

• To create domain-specific class templates:

  % cd ~ptolemy/src/domains/; mkdir new;
  % cd new; mkdom new

• Creates:

  make.template (!!!)
  .../new/kernel
  NEWDomain.cc (NEWDomain and default NEWTarget)
  NEWGeodesic.h
  NEWPortHole.h  NEWPortHole.cc
  NEWScheduler.h  NEWScheduler.cc
  NEWStar.h   NEWStar.cc
  NEWWormhole.h  NEWWormhole.cc
               (NEWWormhole and NEWTo/NEWFromUniversal)

  .../new/stars
  NEWNop.pl

• Compiles (!!!), but does little more.
The Process... an Overview (cont.)

- Create few basic Stars for the domain for testing.
  - debug basic semantics.
- Build up and debug a library of domain-specific Stars.
- Debug heterogeneous interactions.
The Process... an Overview

• Scope out the computational specialization you need.
  • what rules regulate the firing of actors?
  • what rules dictate how/when data is transferred between actors?
  • code generation domains - what supervisory functions are required?
  • are new data types needed?

• Create outlines for new domain classes (see next slide).

• Map the functionality onto domain-specific classes derived from kernel classes.
  • need to understand and accommodate mismatches
  • create any new classes required to support the new semantics

• Consider semantic mapping issues between domains, too.
  • how will/should the domain interact with others?
  • what semantics are built in vs. left for designer?
More Domain-Specific Classes

- EventHorizon
  - ToEH
  - PortHole
  - FromEH

- Kernel
  - XPortHole
  - YPortHole

- Domain
  - InXPortHole
  - OutYPortHole

- XToUniv.
- YFromUniv.

- XWormhole
The Ptolemy kernel defines a set of C++ object classes that implement key abstractions common to all design systems.

Objects derived from these base classes will interoperate in predictable ways.

The key abstractions built into the kernel classes include mechanisms for:

- managing the execution thread of control within the design
- transferring data between actors
- maintaining a consistent picture of time across design elements with potentially very differing concepts of time and how time progresses
Alternatives to Defining an Entirely New Domain

• Derive from a existing Domain.
  • efficient way to customize existing computation models
  • example: BDF and DDF leverage from SDF

• Create a new Target.
  • use an existing code generation Domain with a Target specialized to your application/architecture
Motivations for Defining a New Domain

• Create a tool based on a specialized model of computation.
  • more natural design
  • efficient simulation
  • code generation for simulation acceleration or prototyping hardware and software

• Leverage from an established tool framework.

• Gain interoperability with other Ptolemy domains.
  • support heterogeneous design
  • utilize domain libraries for analysis, display, etc.
Extending Ptolemy with New Domains

Dave Wilson
Berkeley Design Technology, Inc.
(510) 791-9100
wilson@bdti.com