Model-based Operational Control Methods for Smart Manufacturing Systems

Timothy Sprock

Conrad Bock

Systems Integration Division,

National Institute of Standards and Technology,

Gaithersburg, MD

Why is good “scheduling” so hard?

Implementing smart operational control systems is often hampered by:

- Heterogeneous information sources
- Heterogeneous decision support tools
- Heterogeneous execution mechanisms (shop floor “actuators”)

Information Sources:
Part
Process
Resource
Planning
Orders
Shop Status

Decision Support:
Priority Rules
Scheduling Software
    MES? MRP?
Optimization Methods
    Static? Dynamic? Robust?
Simulation

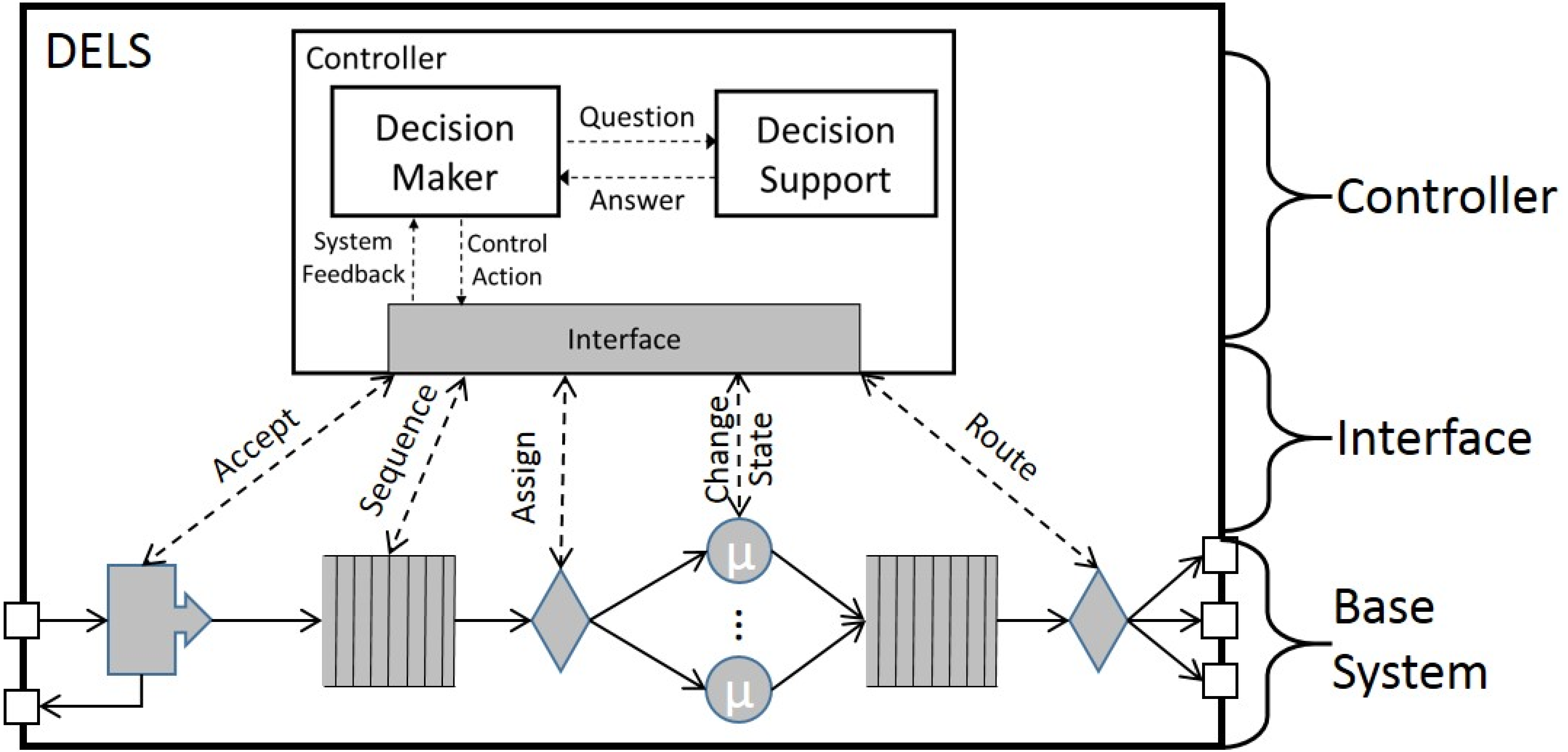
Execution:
Robotic Arms
Conveyors
Overhead Transports
Automated Guided Vehicles (AGV)
Humans

Model-based Operational Control

Three components of successful development and deployment of model-based operational control:

- **Standard model of operational control**
- Analysis models and tools properly implementing that standard (interoperability)
- System-analysis integration methods providing automated, inexpensive access to those analysis tools

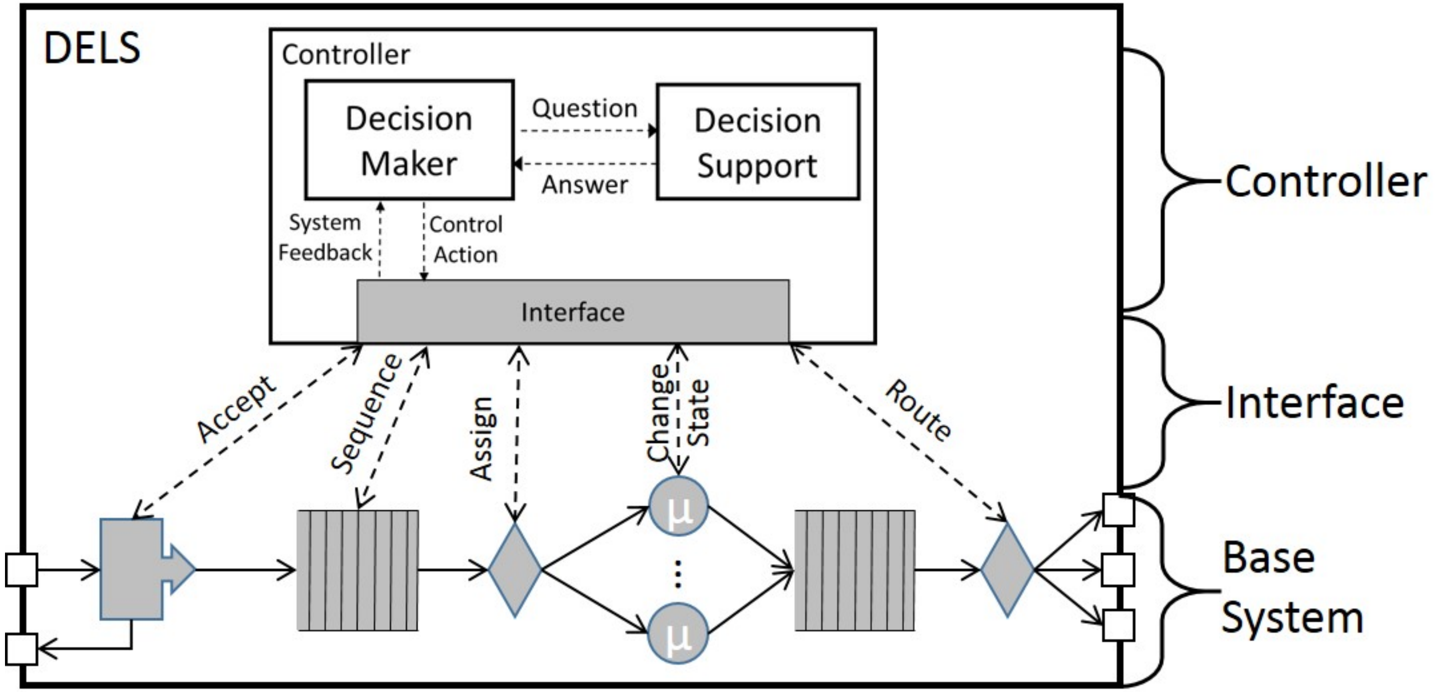
Operational Control Model Overview



Goal: Standard way of describing the base system and operational control of each "functional unit"

Operational Control Model Overview

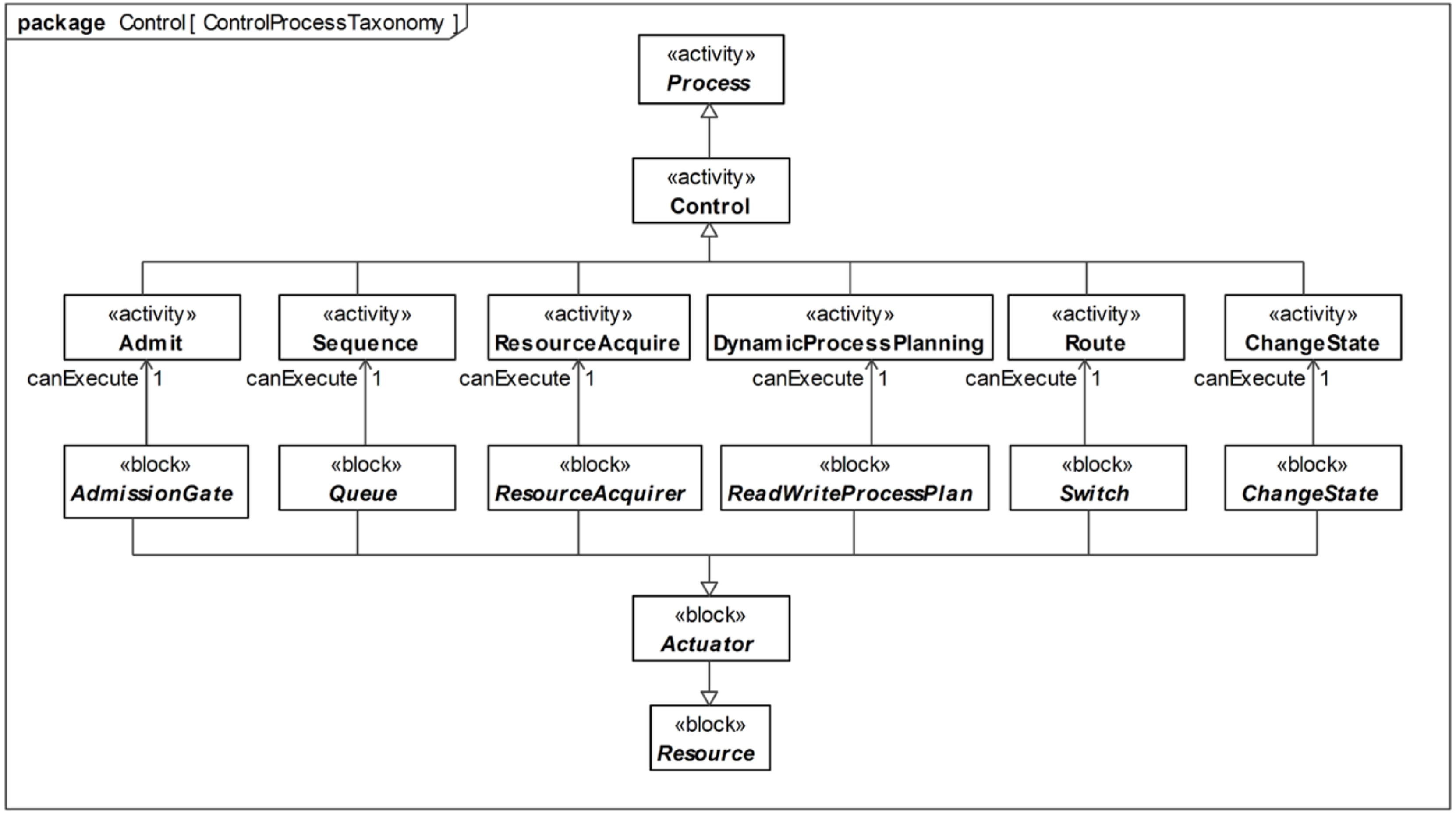
Manipulating flows of tasks and resources through a system.



- Which tasks get serviced? (Admission/Induction)
- When {sequence, time} does a task get serviced? (Sequencing/Scheduling)
- Which resource services a task? (Assignment/Scheduling)
- Where does a task go after service? (Routing/Dynamic Process Planning)
- What is the state of a resource? (task/services can it service/provide)

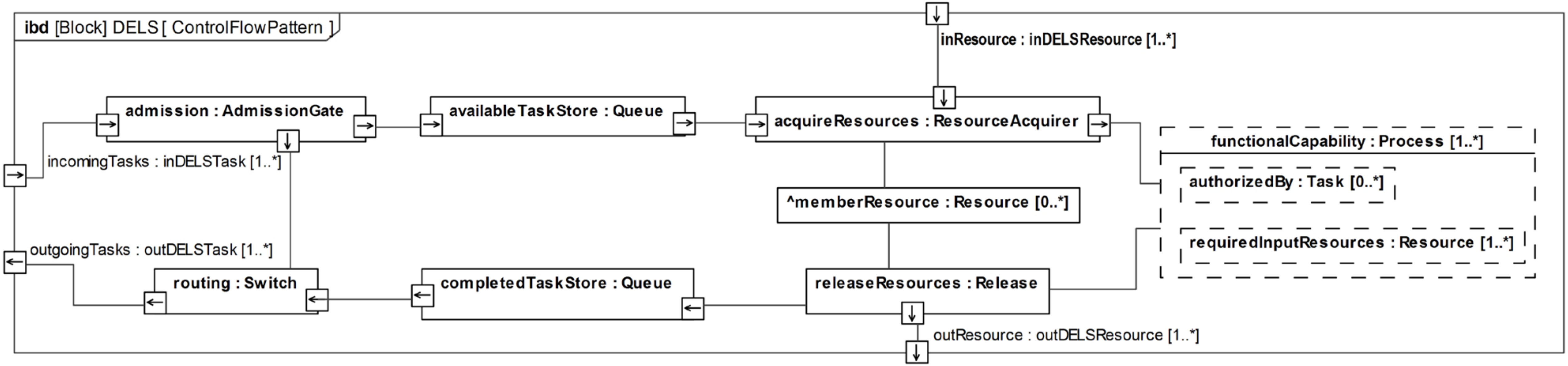
Operational Control Model Library

Functional Capabilities and Resource Roles: Building blocks for assembling models of system capable of implementing operational control



Reference Patterns

Templates guide implementation of operational control building blocks in a system design

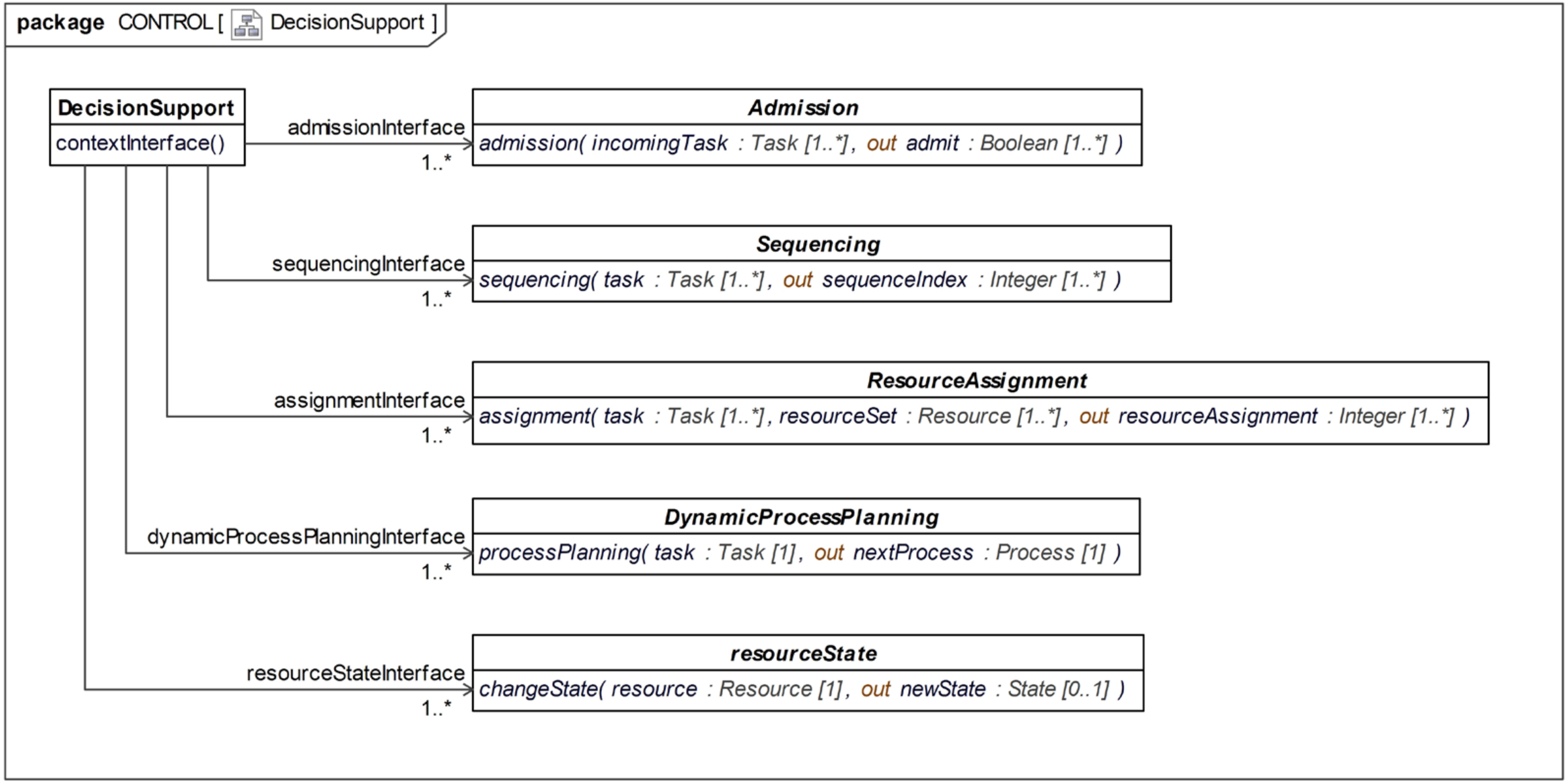


For existing systems, patterns guide discussions on how operational control works: Where/when is the decision made? How is it made? How is the control decision executed?

Pseudo-checklist for system designers to provide resources (system objects) to fulfill these operational control roles.

Standard Decision-support Interfaces

Controllers are configured with algorithms that provide decision support for each control decision

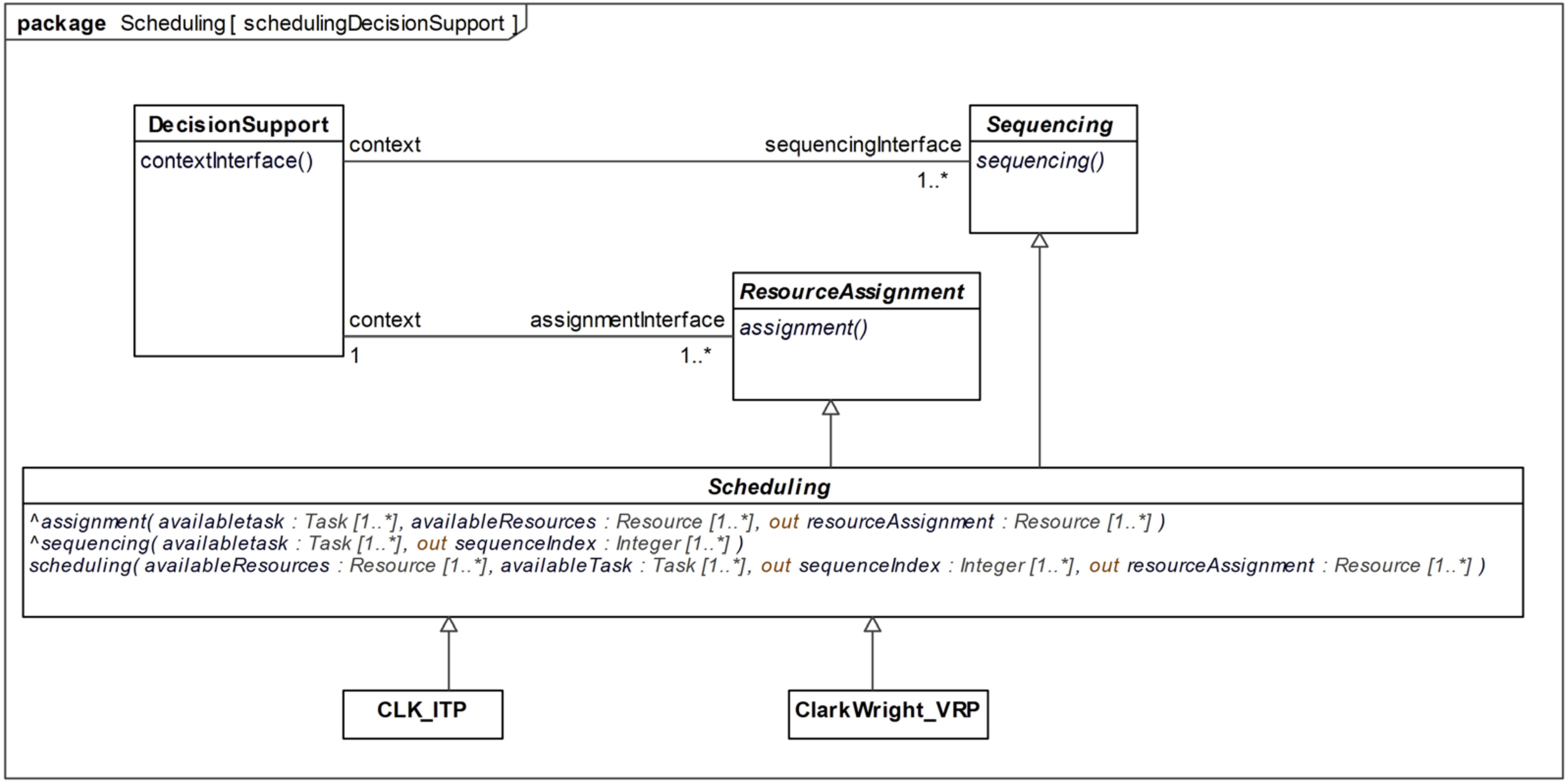


Patterns for Modeling Operational Control

Link decision support in the controller to behaviors and actuators on the shop floor

	Sequencing	Assignment
Question	"In what order {sequence, time} should tasks be served?"	"Which resource is assigned to service the task?"
Decision Function	$\mathit{Sequence}\text{: Task} \rightarrow \mathbb{N}$	$\mathit{Assign}\text{: Task} \times \text{Resource}(s) \mapsto \text{Resource}(s)$
Actuator Function	$\text{Sequence(TaskSet)} := \text{sort(TaskSet, sequenceIndex)} = \text{TaskSet}'$	$\mathit{Assign}\text{(Task, Resource)} :=$ $\text{Task.nextProcessStep.requiredInputResource} \leftarrow \text{Resource}$
Decision Expression	$x_{lk} = 1, \quad \text{if task } l \text{ is serviced } k^{th}$	$x_l^m = 1$ if resource m is assigned to execute the next process step of task l $x_{lj}^R = 1$ if resource group R is assigned to execute the j^{th} process step of task l
Decision Support Interface	«Strategy» ***Sequencing*** *sequencing(out sequenceIndex : Integer [1..*], taskSet : Task [1..*])*	***ResourceAssignment*** *assignment(availabletask : Task [1..*], availableResources : Resource [1..*], out resourceAssignment : Resource [1..*])*
Actuator Function – System Model Library Component	availableTasks : Task[1..*] sequenceIndex : Integer[1..*] **: Sequence** orderedTasks : Task[1..*]	resourceAssignment : Integer[1..*] task : Task[1] **: ResourceAcquire** acquiredResources : Resource[1..*] availableResource : Resource[1..*]
Actuator – System Model Library components	inTask : inDELSTask [1..*] «block» ***Queue*** outTask : outDELSTask [1..*]	inTask : inDELSTask [1] availableResource : inDELSResource [1..*] «block» ***ResourceAcquirer*** acquiredResource : outDELSResource [1..*]

Goal: Standard Interface Enables Interoperability



Example: “Adapt” Algorithms to Interface

Existing analysis models, such as those for scheduling, don’t naturally conform to the standard and need to be adapted to become “plug-and-play”

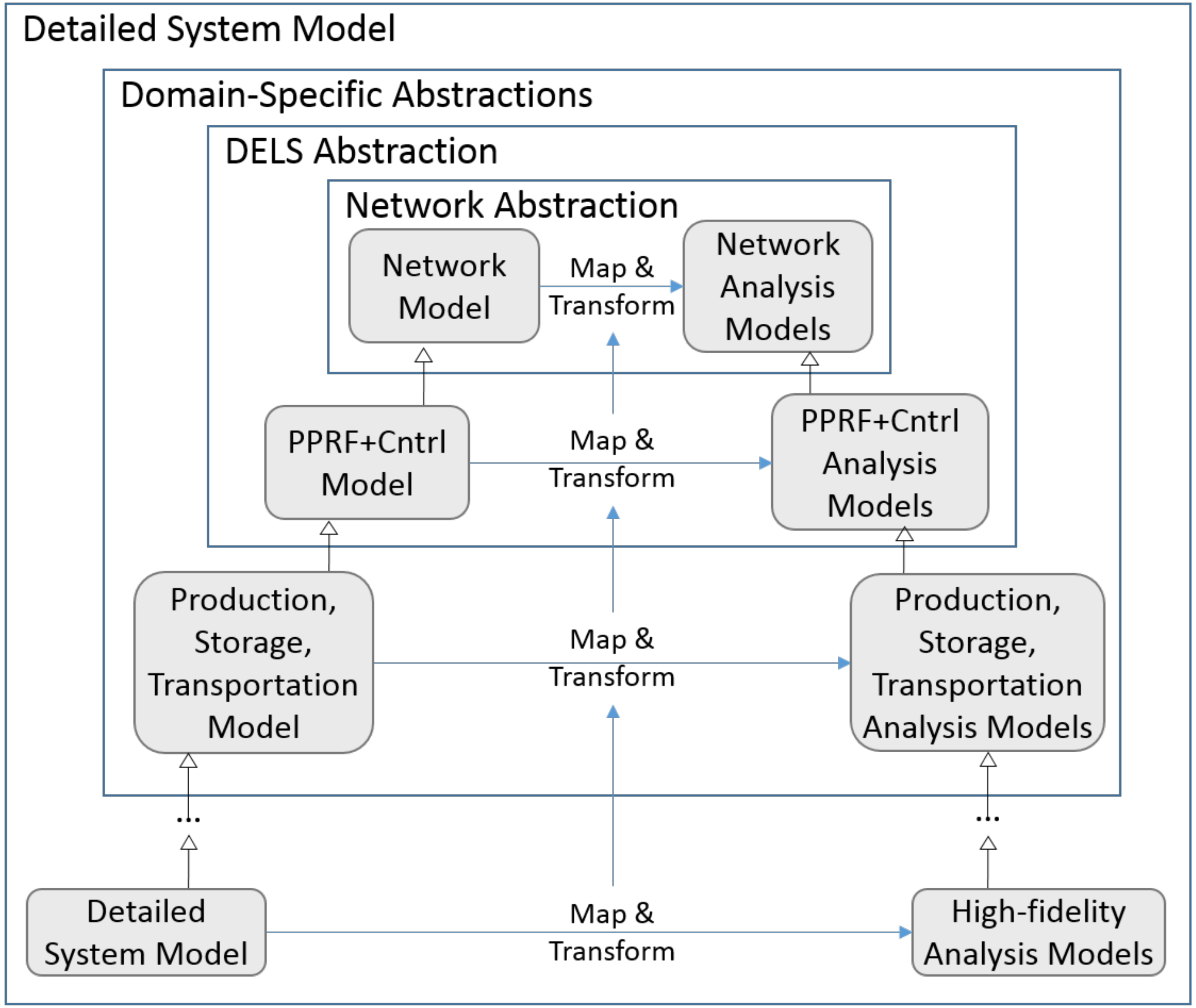
```
classdef Scheduling_ClarkWrightVRP < SchedulingInterface
    %implements SCHEDULING_STRATEGY
    %using Clark Wright VRP algorithm

    properties
    end

    methods
        function Scheduling(self, TaskList, ResourceSet)

         %% 1. Add Depot %%...%%
         %% 2. Make Cost Matrix & Capacity Array %%...%%
         %% 3. Call Clark Wright Savings  %%...%%
         %% 4. Assign ordered tasklists to resources
            for j = 1:min(length(ResourceSet), length(loc))
                ResourceSet(j).TaskList = TaskList(loc{j});
                ResourceSet(j).TaskListCapacityRequirement = TC(j);
            end
        end
    end

end
```

```
classdef Scheduling_CLK_ITP < SchedulingInterface
    %Implements SCHEDULING_STRATEGY
    %Using Iterative Tour Partition on Chained-LK TSP solution

    properties
    end

    methods
        function Scheduling(self, TaskList, ResourceSet)
            %Using Iterative Tour Partition on Chained-LK TSP solution
            %% 1. Make Cost Matrix %%...%%
            %% 2. Initialize with Nearest Neighbor Heuristic %%...%%
            %% 3. Call chained_lin_kernighan %%...%%
            %% 4. Add Depot %%...%%
            %% 5. Partition the TSP tour %%...%%
            %% 6.Assign ordered tasklists to resources
            for j = 1:length(self.Controller.DELS.ResourceSet)
                ResourceSet(j).TaskList = TaskList(partitions(2:end-1,j));
            end
        end
    end
end
```

System-Analysis Integration Methods

- Use a common representation of the system under control (system model) to integrate multiple sources of information already defined and/or represented in other ways, often from heterogenous systems in incompatible formats, to create an integrated model of the system.

- Integrate system models with many kinds of analysis models, such discrete event simulation.



Goal: Enable Simulation-based Methods

- [Design] Standard system models and supporting analysis methods will enable simulation-based methods to be routinely applied during the (re-)design process to test and validate control logic in high-fidelity simulations before deploying to the system

- [Operation] Simulation can also be integrated with optimization and heuristic methods to provide online decision support

- [Goal] Pathway from design and analysis of control to testing, validation, and deployment.

Need for Model-Based Methods for Smart Manufacturing

- Current methods and tools are limited for production systems engineering
  - Formal specification & analysis automation
  - Design and teaching

- Documentation & Organization of Knowledge
  - Existing Systems Models (industry)
  - Existing Analysis Models (academia)

- Bridge between system and analysis models
  - Interoperability between different analysis models of the same system
  - Greater reusability of analysis: collaboration and automation
  - Modeling & Simulation Interoperability (MSI)

Model-based Operational Control

Challenges:

- Heterogeneous information sources
- Heterogeneous decision support tools
- Heterogeneous execution mechanisms (shop floor “actuators”)

Approach:

- Standard model of operational control
- Analysis models and tools properly implementing that standard (interoperability)
- System-analysis integration methods providing automated, inexpensive access to those analysis tools

<u>Information Sources:</u>
Part
Process
Resource
Planning
Orders
Shop Status

<u>Decision Support:</u>
Priority Rules
Scheduling Software
  MES? MRP?
Optimization Methods
  Static? Dynamic? Robust?
Simulation

<u>Execution</u>:
Robotic Arms
Conveyors
Overhead Transports
Automated Guided Vehicles (AGV)
Humans

More Information

- System-Analysis Integration (SAI) Project
  - Conrad Bock, project lead conrad.bock@nist.gov
  - http://www.nist.gov/el/msid/syseng/smsi.cfm
- Discrete event logistics systems (DELS)
  - Tim Sprock, timothy.sprock@nist.gov
  - INCOSE Production and Logistics Systems Modeling Challenge Team
    - http://www.omgwiki.org/MBSE/doku.php?id=mbse:prodlog