

Introduction to PSL

Conrad Bock
NIST

“PSL” stands for ...

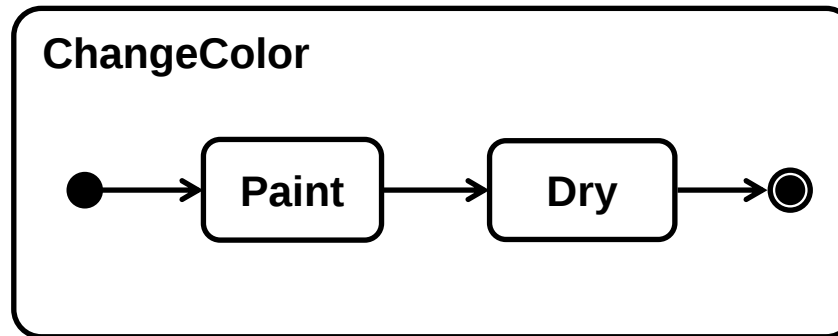
- **... Process Specification Language,**
- **... but not in the sense of a process model or programming language.**
- **Should be called:**
 - **Process Semantics Language.**
 - **Process Constraint Language.**
- **PSL enables:**
 - **Describing what actually happens when a process specification executes.**
 - **Writing constraints on processes.**

PSL Background

- **Based on long period of research:**
 - Situation calculus.
 - Process Interchange Format (PIF).
 - Enterprise modeling.
- **Applied to scheduling, process modeling, process planning, production planning, simulation, project management, workflow, business process reengineering, vehicle navigation, semantic interoperability.**
- **Project 18629 at the ISO.**
- **Basis of Semantic Web Services (SWSL) at W3C.**
 - “Some philosophical problems from the standpoint of artificial intelligence,” McCarthy, J., Hayes, P., in Meltzer B, Michie D (eds) Machine Intelligence 4, Edinburgh University Press, Edinburgh, pp 463–502, 1969.
 - Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems, Reiter, R., MIT Press, 2001.
 - The Process Interchange Format Project, <http://ccs.mit.edu/pif>.
 - “The TOVE Project: A Common-sense Model of the Enterprise, Industrial and Engineering Applications of Artificial Intelligence and Expert Systems,” Fox, M., in Bellif, Radermacher, F. (eds.), Lecture Notes in Artificial Intelligence # 604, Springer-Verlag, pp 25–34, 1992.
 - “Enterprise Modelling,” Fox, M., Gruninger, M., AI Magazine, AAAI Press, pp. 109–121, Fall 1998.
 - Semantic Web Services Framework (SWSF), W3C, <http://www.w3.org/Submission/2005/07/>, 2005.

Process Models

- UML 2:



(or UML repository)

- BPEL:

```
<process name="ChangeColor">
  <sequence>
    <invoke operation="Paint"></invoke>
    <invoke operation="Dry"></invoke>
  </sequence>
</process>
```

- C:

```
void ChangeColor
{ Paint();
  Dry();
}
```

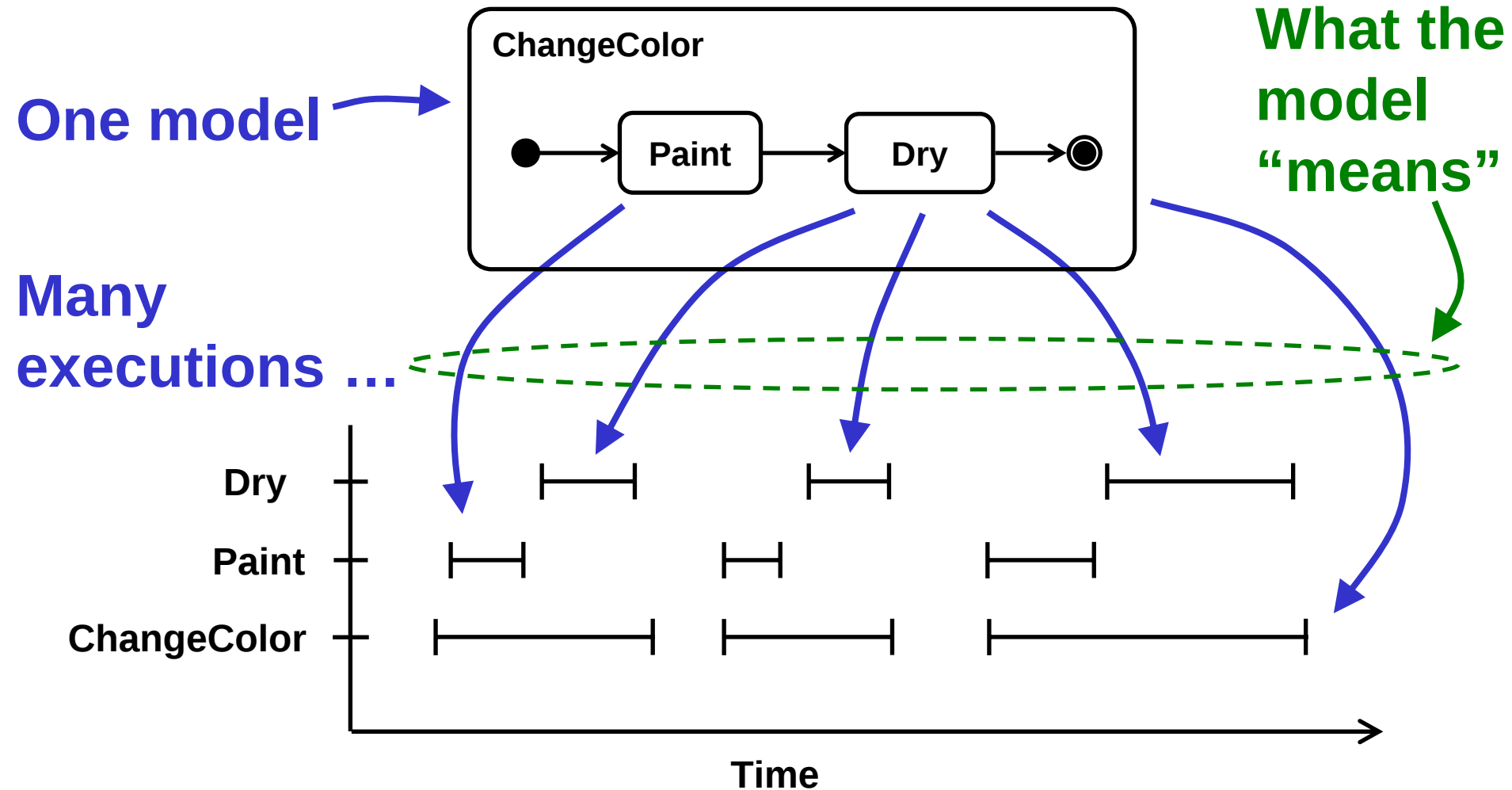
What Happens?

- **Does Dry start only after Paint finishes?**
 - Yes, unless Paint is invoked asynchronously.
- **Does Dry happen everytime Paint does?**
 - Not necessarily, the model is referring only to the ChangeColor process.
- **These questions are about the actual execution of the process.**
 - Which steps start and stop when.
 - What the process model “means”.

Capture the Meaning

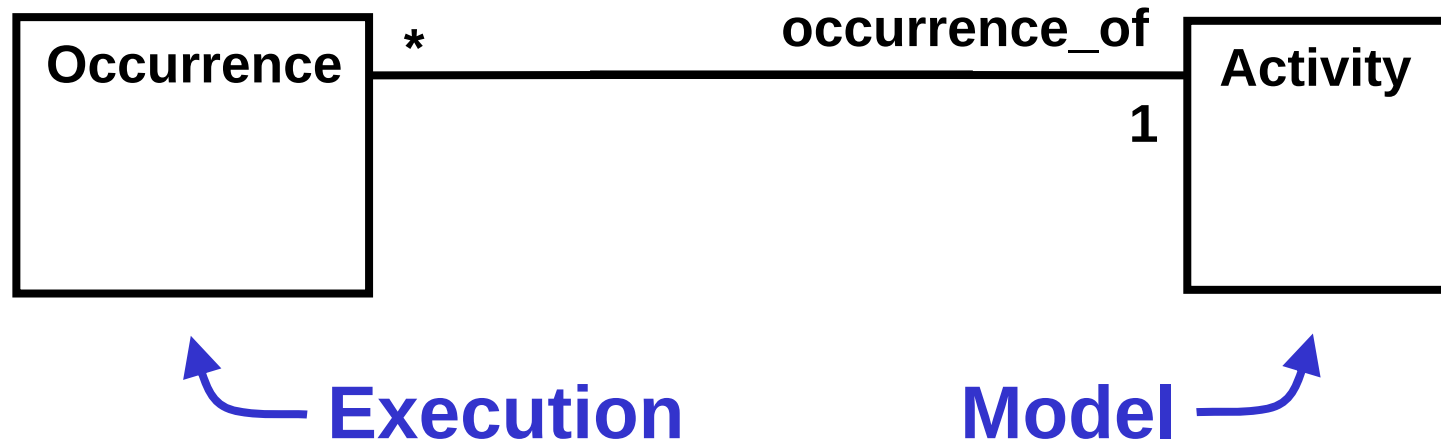
- **How do we know what the model means?**
 - Read the language documentation.
 - Execute the model on a reference implementation.
 - Ask somebody.
- **Humans eventually figure it out, but what about the tools?**
 - They can't read documentation, experiment with implementations, or ask anyone.
- **Need a way to express the meaning of process models in a way tools can understand.**

Model vs. Execution



- ... each satisfying the constraints of the model.

PSL Model of Execution



- **Occurrence is an execution of an Activity**
 - for example, Paint executed at 10:22am ET 9/1/2003 at factory 1.

PSL Model of Execution

- PSL is defined in the Common Logic Interchange Format (CLIF)...
- ... but could be OCL or other FOL language.

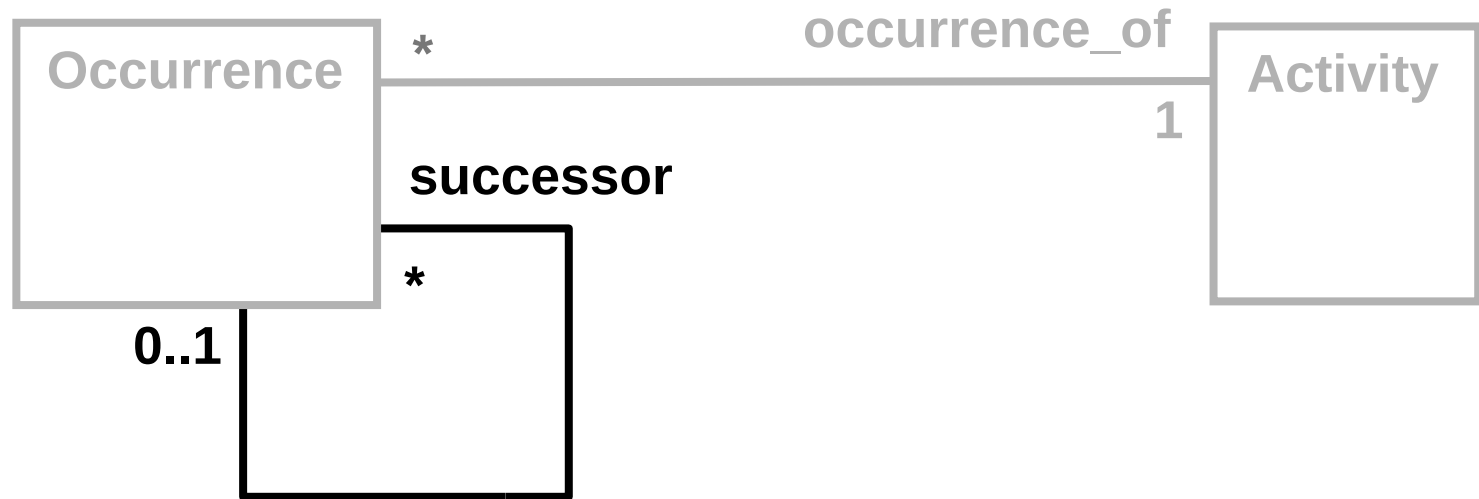
```
(forall (?occ ?a)
  (implies (occurrence_of ?occ ?a)
    (and (activity_occurrence ?occ)
      (activity ?a))))

(forall (?occ)
  (implies (activity_occurrence_of ?occ)
    (exists (?a)
      (and (activity ?a)
        (occurrence_of ?occ ?a)))))

(forall (?occ ?a1 ?a2)
  (implies (and (occurrence_of ?occ ?a1)
    (occurrence_of ?occ ?a2)
    (= ?a1 ?a2))))
```

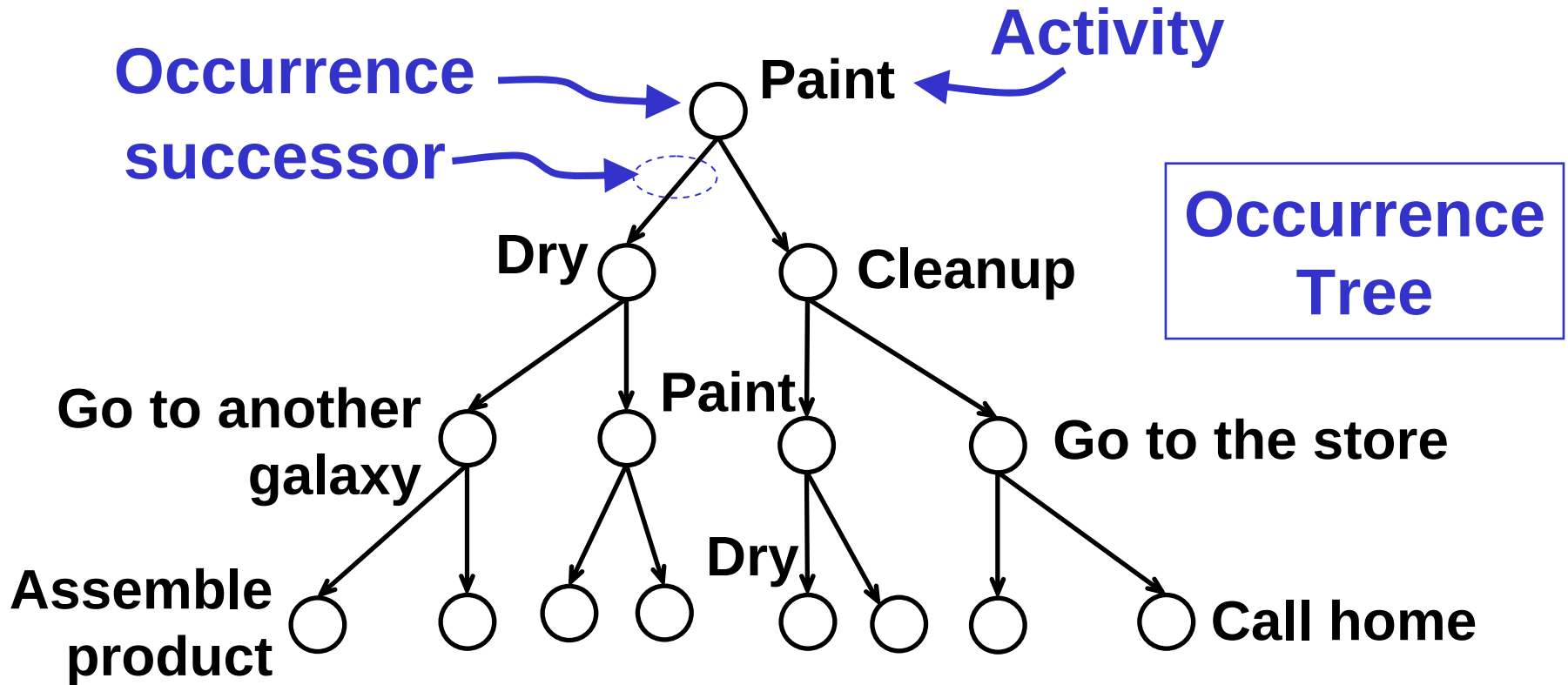
- "Common Logic (CL) - A Framework for a Family of Logic-Based Languages," ISO, WG2, SC32, IEC JTC1, <http://philebus.tamu.edu/cl>, June 2005.
- "Object Constraint Language," OMG, <http://doc.omg.org/formal/06-05-01>, May 2006.

PSL Model of Execution



- **Executions happen one after another.**
- **Notice the multiplicities:**
 - **Occurrence has multiple successors, one for each (theoretically) possible next occurrence.**

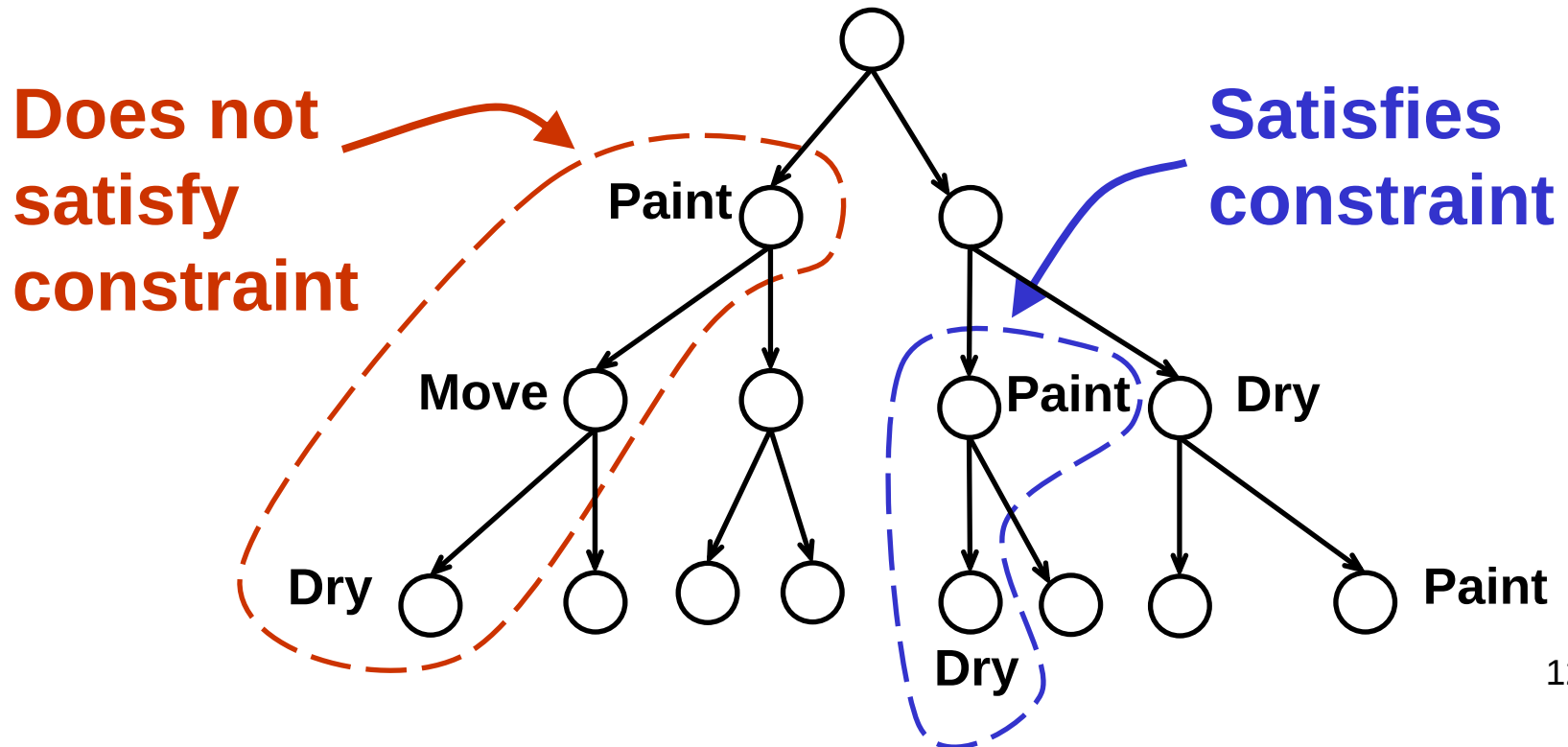
Anything Can Happen



- Tree of all possible execution sequences, including
 - not physically possible.
 - not specified by the user.
- Not stored anywhere, just referred to by constraints¹¹

Rules as Occurrence Constraints

- Specify which parts of the occurrence tree are “legal”.
- Example rule: drying immediately follows all painting.

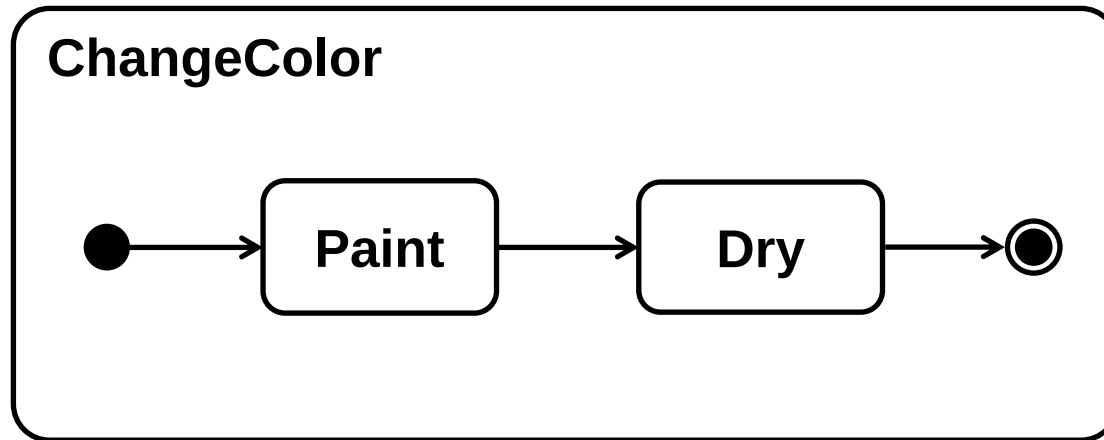


Constraint Language

- In CLIF:

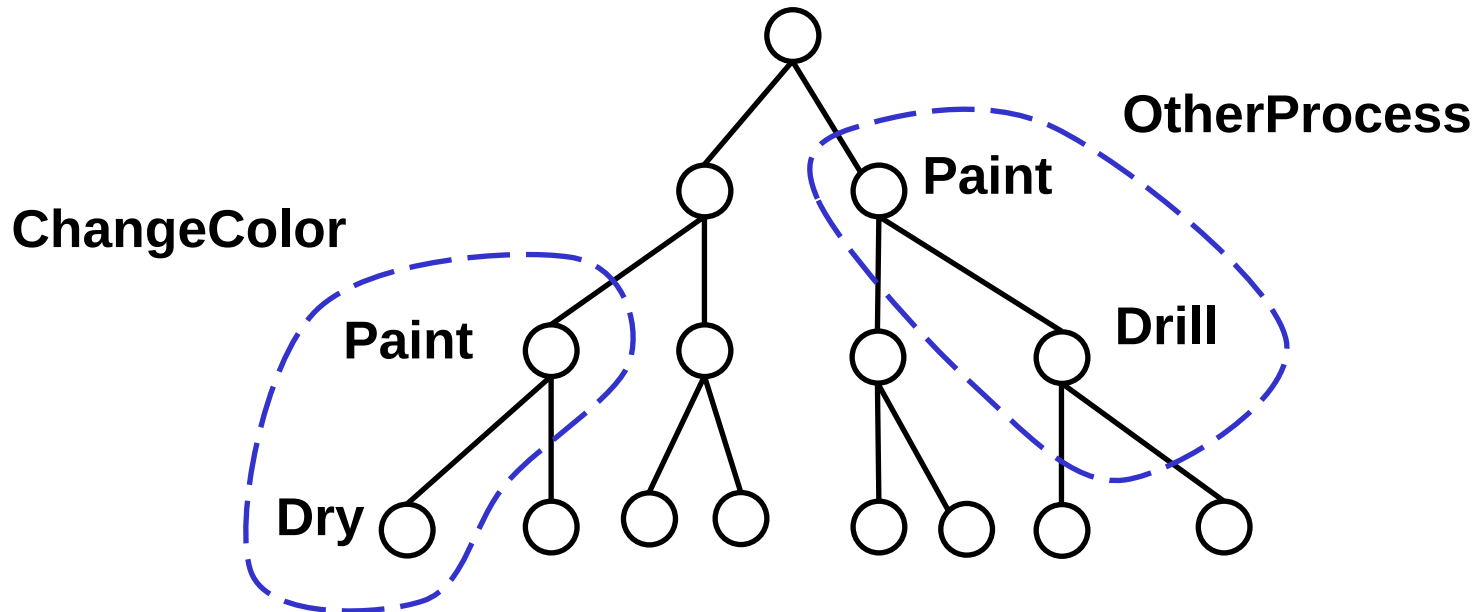
```
(forall (?occPaint)
  (implies
    (and (occurrence_of ?occPaint Paint)
          (legal ?occPaint))
    (and (legal (successor Dry ?occPaint))
          (forall (?otherSuccessor)
            (implies
              (not (equal ?otherSuccessor
                           (successor Dry ?occPaint)))
              (not (legal ?otherSuccessor))))))))
```

Processes in PSL



- Above says that Dry happens after Paint *under executions of ChangeColor*.
- Outside of ChangeColor Paint can occur without Dry.

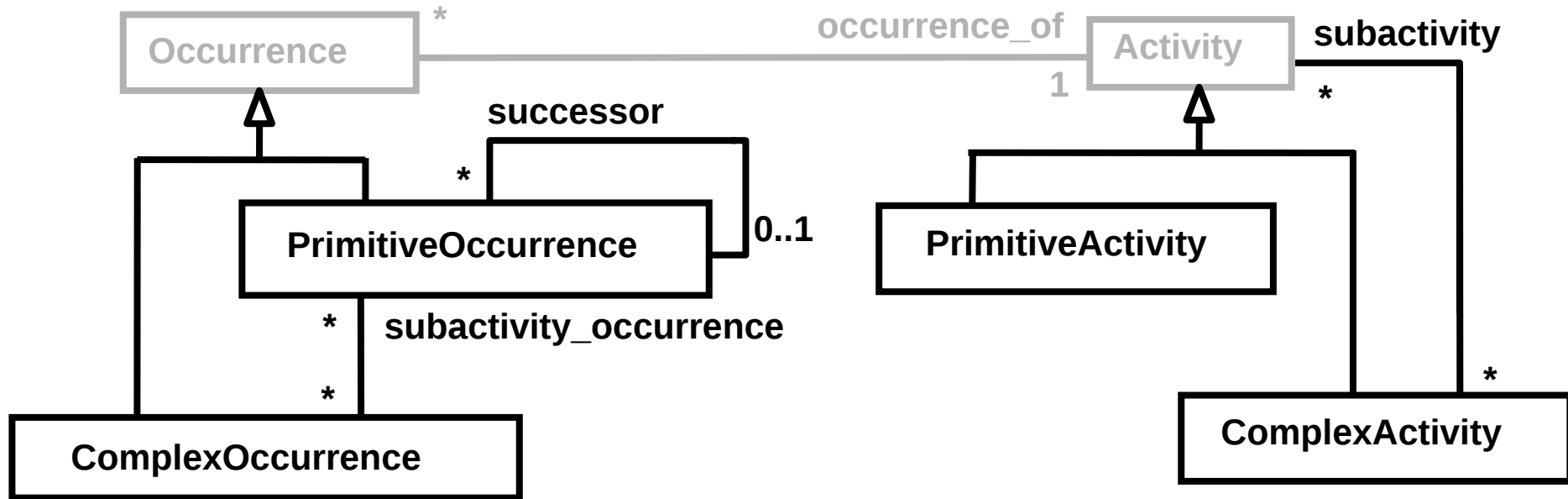
Processes as Occurrence Constraints



- **Paint happens immediately after Dry under executions of ChangeColor.**
- **ChangeColor specification does not constrain OtherProcess above**

Complex Processes in PSL

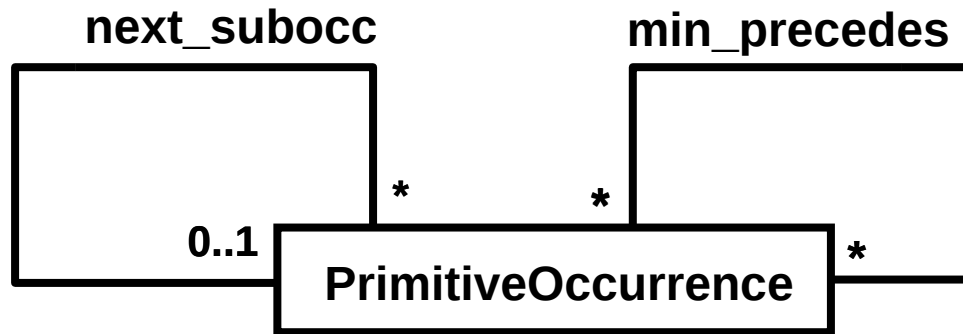
- Complex occurrences and activities composed of primitive ones:



- Successor moved to PrimitiveOccurrence.
- Occurrence tree covers every step at finest level of granularity.

Complex Processes in PSL

- Execution sequencing within complex activity:



Occurrences following *immediately*

Occurrences following *sometime*, not necessarily immediately.

- (Defined in terms of successor)

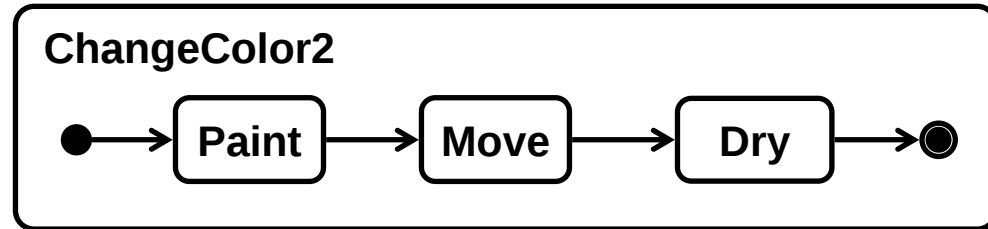
Complex Processes in PSL

- **Constrain occurrences of ChangeColor to be composed of sequential occurrences of Paint and Dry:**

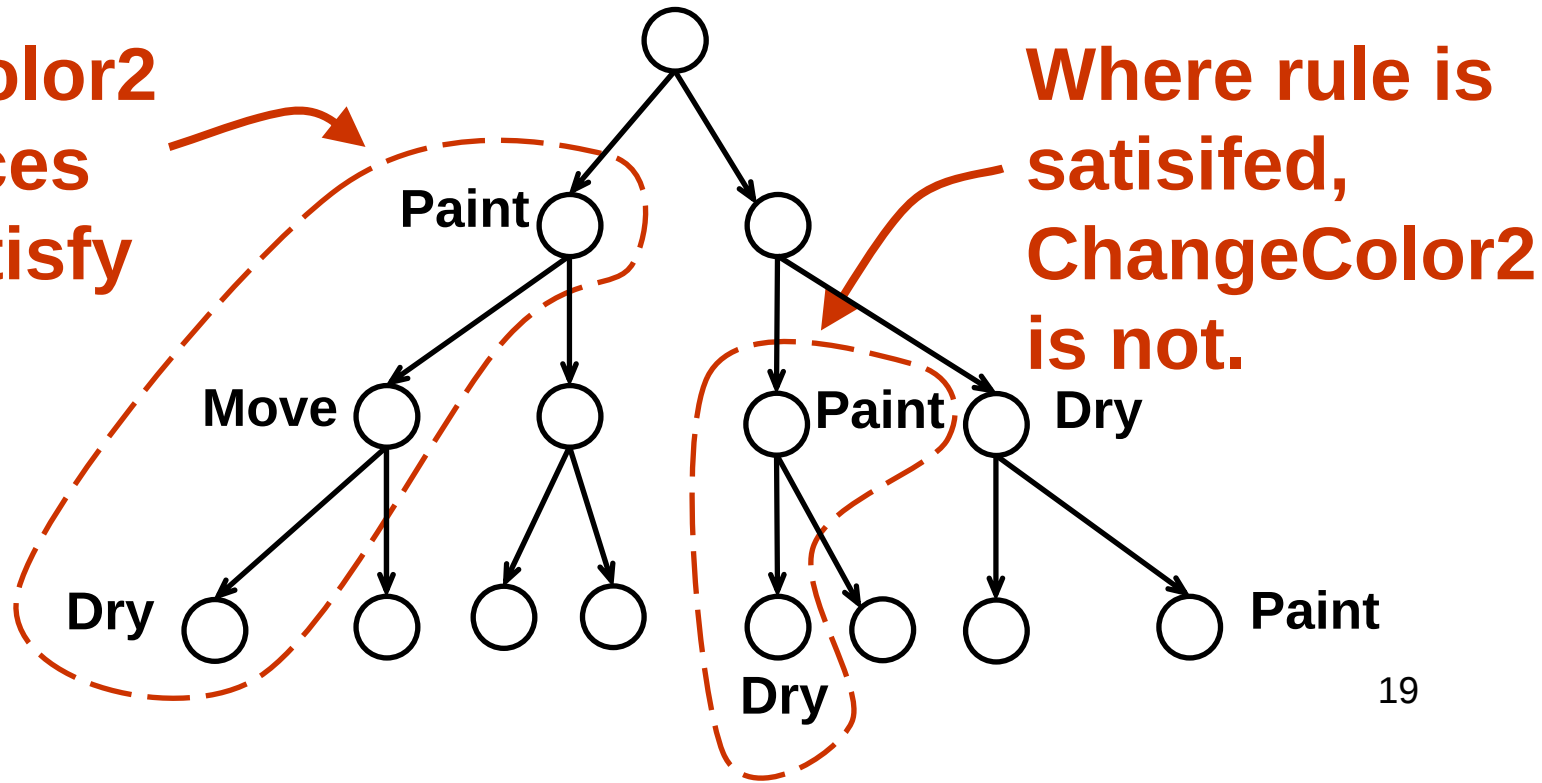
```
(forall (?occChangeColor)
  (implies
    (occurrence_of ?occChangeColor ChangeColor)
    (exists (?occPaint ?occDry)
      (and (occurrence_of ?occPaint Paint)
            (occurrence_of ?occDry Dry)
            (subactivity_occurrence ?occPaint ?occChangeColor)
            (subactivity_occurrence ?occDry ?occChangeColor)
            (next_subocc ?occPaint ?occDry
                          ChangeColor))))))
```

Rule / Process Consistency

- Rule: drying immediately follows all painting.
- Process:

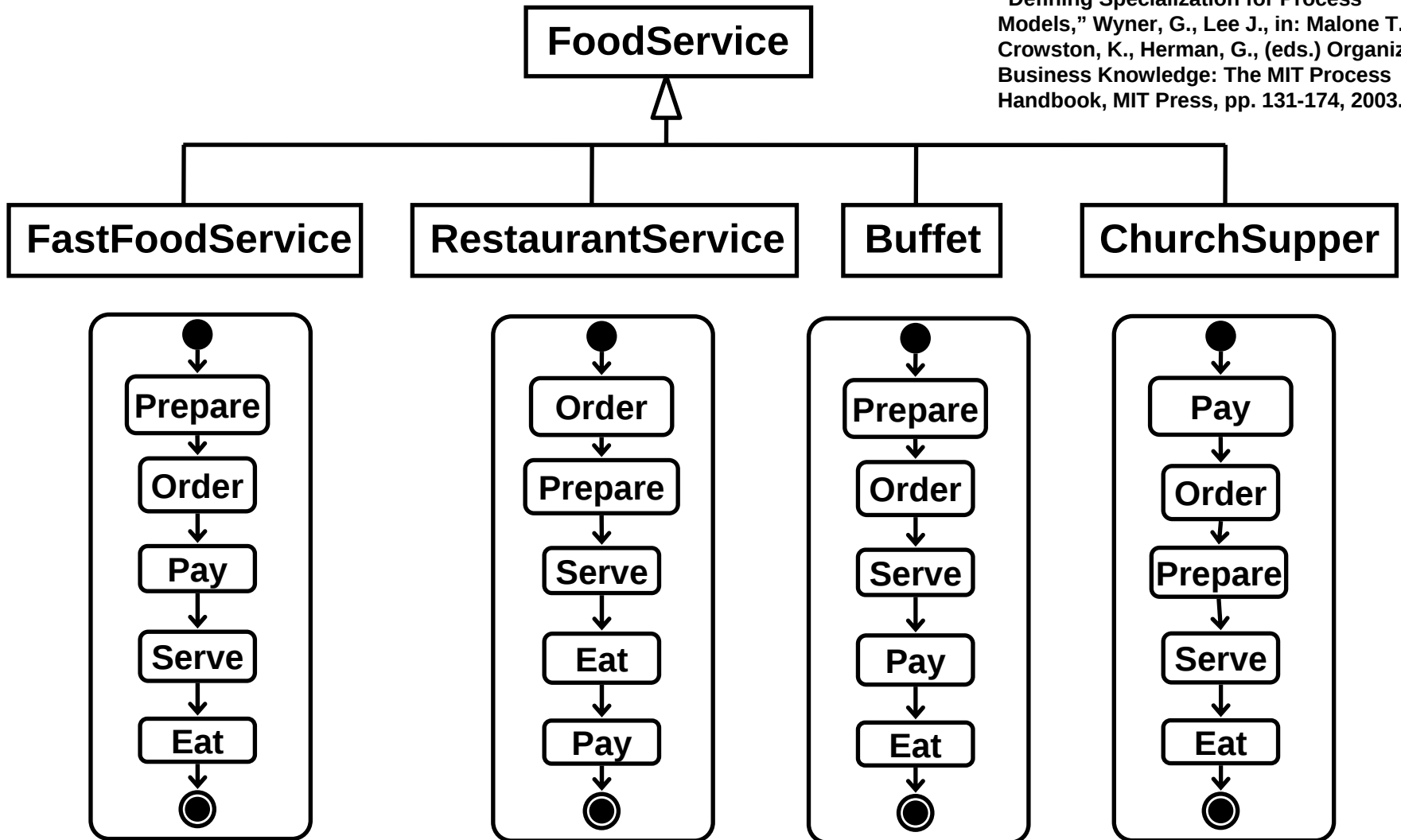


ChangeColor2 occurrences do not satisfy rule.



Behavior Classification

“Defining Specialization for Process Models,” Wyner, G., Lee J., in: Malone T., Crowston, K., Herman, G., (eds.) Organizing Business Knowledge: The MIT Process Handbook, MIT Press, pp. 131-174, 2003.



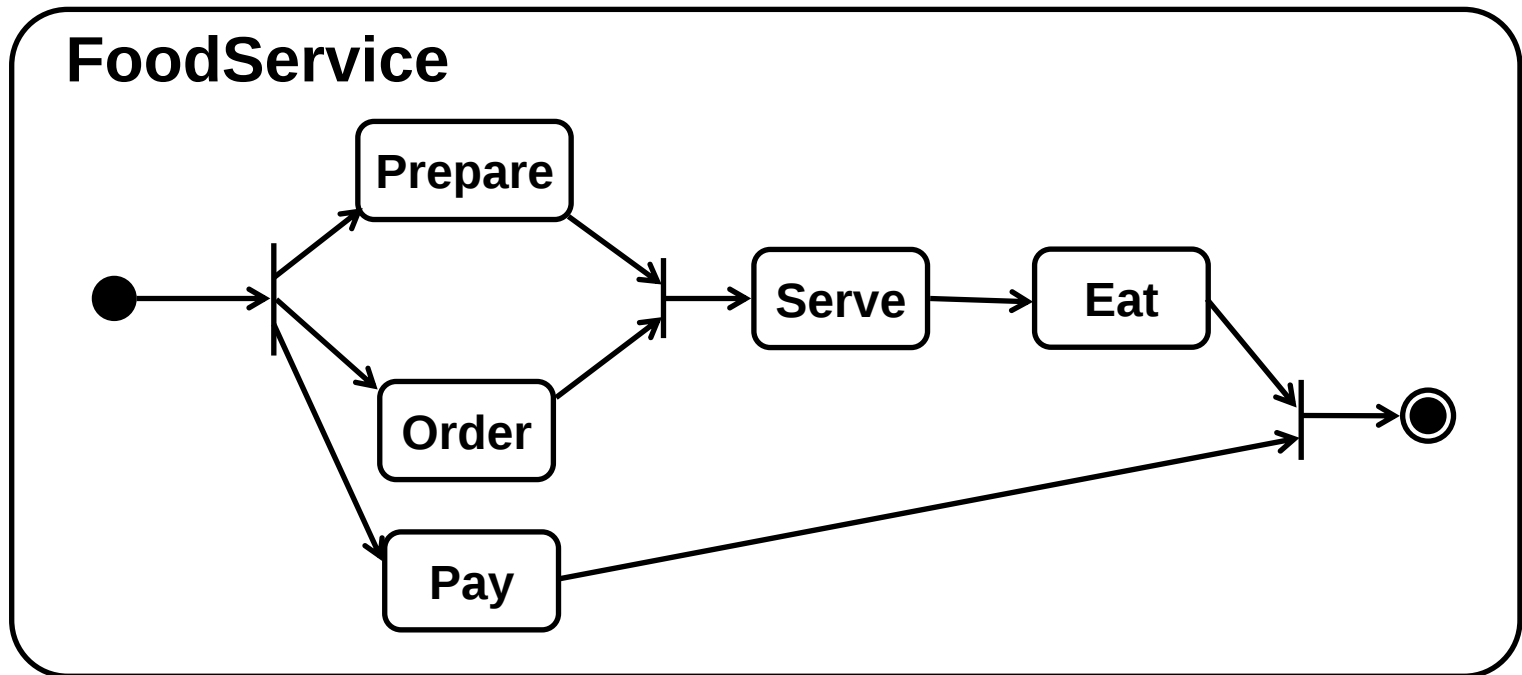
- How to abstract commonality?

Behavior Classification

- **Food Service has these steps:**
 - Order, Prepare, Serve, Eat, Pay
- **With these constraints:**
 - Order, Prepare, and Serve always happen before Eat.
 - Serve happens after Prepare and Order.
 - Pay can happen anytime in the process.
- **Fast Food Service adds:**
 - Prepare before Order.
- **Need to *partially* specify a process ...**
- **... and incrementally build up constraints.**

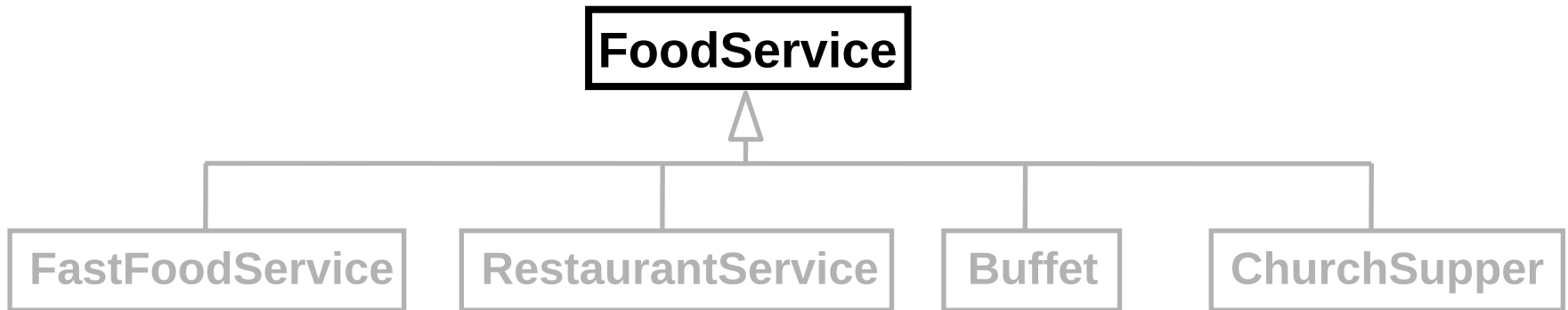
Behavior Classification

- Flow models are not expressive enough:



- Prepare and Order are not concurrent.
- Pay is not concurrent with other steps.

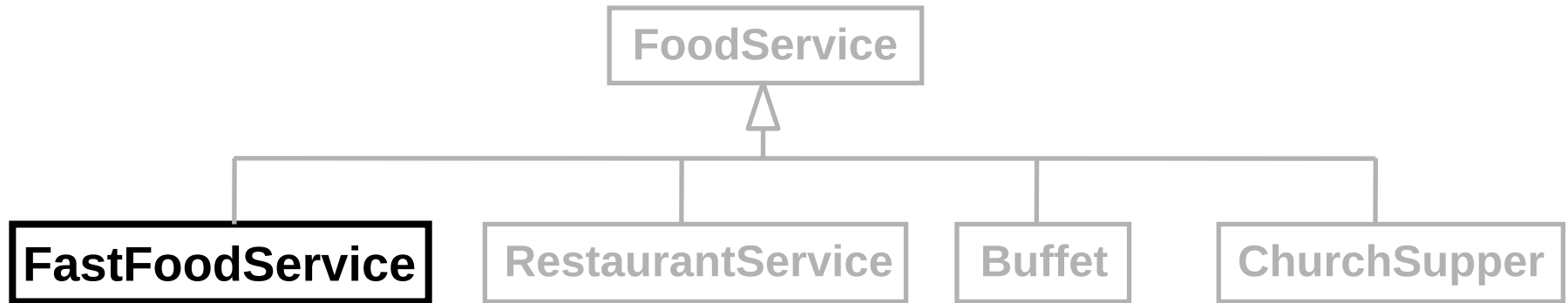
Behavior Classification



- **FoodService: Prepare sometime before Eat.**

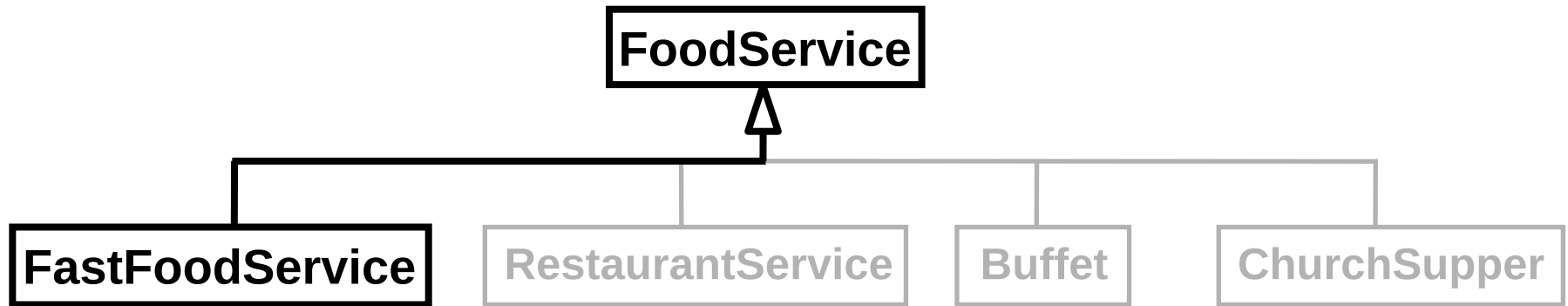
```
(forall (?occFoodService)
  (implies
    (occurrence_of ?occFoodService FoodService)
    (exists (?occPrepare ?occEat)
      (and
        (occurrence_of ?occPrepare Prepare)
        (occurrence_of ?occEat Eat)
        (subactivity_occurrence ?occPrepare ?occFoodService)
        (subactivity_occurrence ?occServe ?occFoodService)
        (min_precedes ?occPrepare ?occEat
          FoodService))))))
```

Behavior Classification



- **FastFoodService: Prepare sometime before Order.**
(**forall** (?occFastFoodService)
 (**implies**
 (**occurrence_of** ?occFastFoodService FastFoodService)
 (**exists** (?occPrepare ?occOrder)
 (**and**
 (**occurrence_of** ?occPrepare Prepare)
 (**occurrence_of** ?occOrder Order)
 (**subactivity_occurrence** ?occPrepare ?occFoodService)
 (**subactivity_occurrence** ?occOrder ?occFoodService)
 (**min_precedes** ?occPrepare ?occOrder
 FoodService))))))

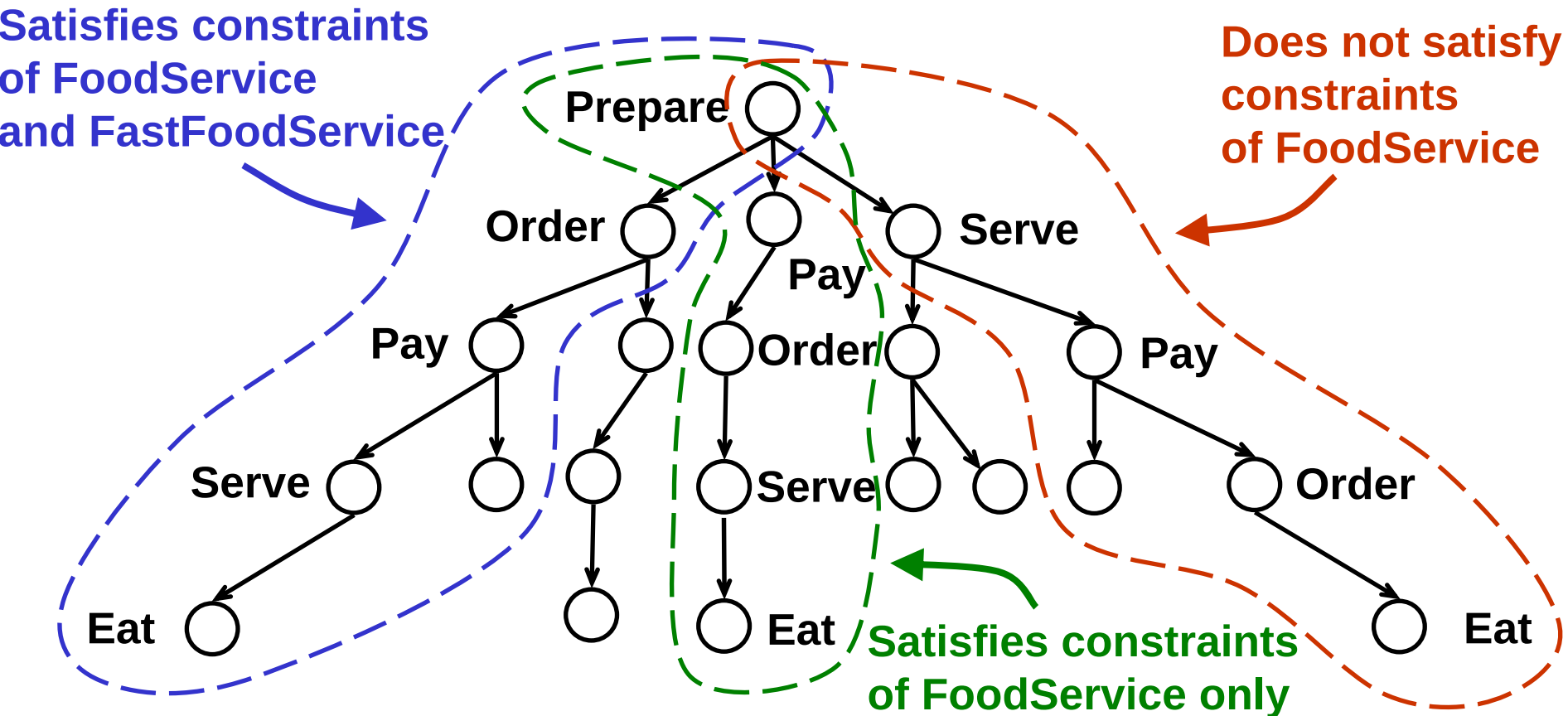
Behavior Classification



- **Classification of process executions:**
 - All subactivity occurrences of FastFoodService occurrences are subactivity occurrences of FoodService occurrences.

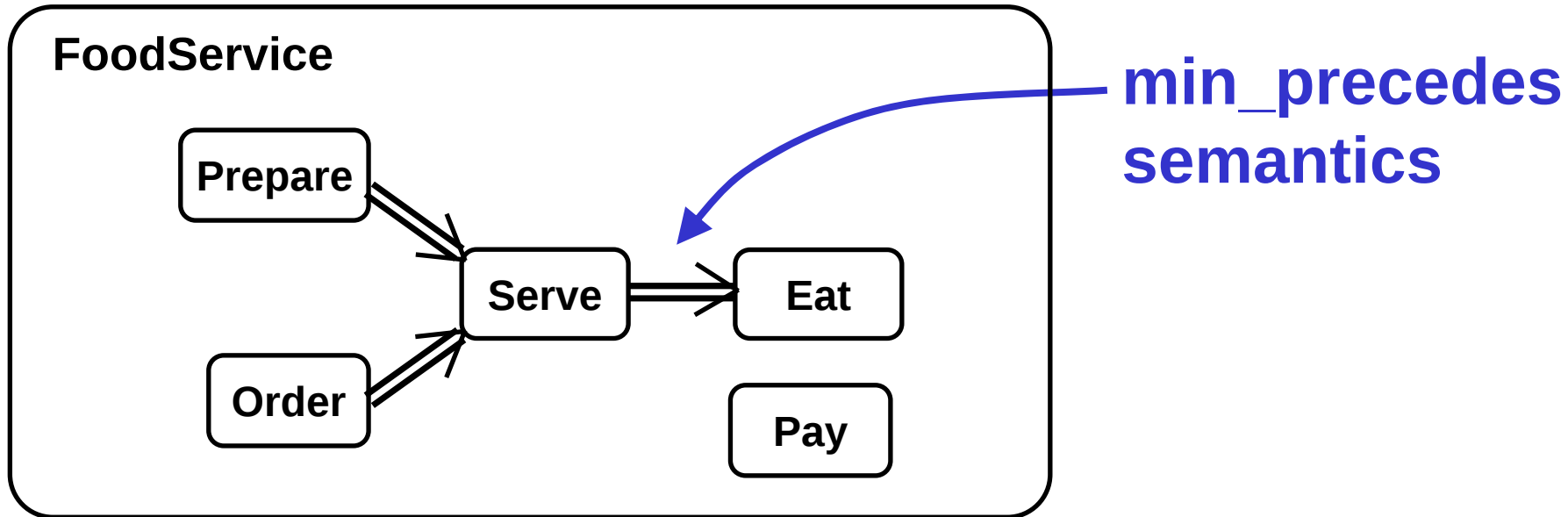
```
(forall (?occFFS)
  (implies (occurrence_of ?occFFS FastFoodService)
    (exists (?occFS)
      (and (occurrence_of ?occFS ?FoodService)
        (forall (?s)
          (implies
            (subactivity_occurrence ?s ?occFFS)
            (subactivity_occurrence ?s ?occFS))))))))25
```

Behavior Classification



- Execution traces conforming to general and/or specific process constraints, or not.

Behavior Classification



- **Possible enhancement to UML notation.**
- **(Requires updating tools and services ...**
- **... compared to extending CLIF representation)**

Web Service Queries

- **Buy a book:**
 - without using a credit card.
 - credit card charged only when shipped.

(adapted from example by Michael Gruninger)
- **Shipping:**
 - transport frozen vegetables from San Francisco to DC.
- **Substituting:**
 - a web service with another that achieves the desired effects at lower cost.

Web Service Queries

- **Web service posts specification of the public aspects of their process.**
- **Query is a specification of the desired aspects of a process.**
- **Answer tells which web service are consistent with the query.**
 - **Which posted specifications are generalizations of the query?**

Web Service Queries

- Buy a book without using a credit card.

```
(forall (?occ)
  (implies (occurrence-of ?occ DesiredProcess)
    (and (exists (?s1)
      (and (occurrence-of ?s1 ShipBook)
        (subactivity-occurrence ?s1 ?occ))
      (not (exists (?s2)
        (and (occurrence-of ?s2 ChargeCreditCard)
          (subactivity-occurrence ?s2 ?occ))))))))))
```

- ... with credit card charged after ship.

```
(forall (?occ)
  (implies (occurrence-of ?occ DesiredProcess)
    (exists (?s1 ?s2)
      (and (occurrence-of ?s1 ShipBook)
        (subactivity-occurrence ?s1 ?occ))
      (occurrence-of ?s2 ChargeCreditCard)
      (subactivity-occurrence ?s2 ?occ)
      (min_precedes ?s1 ?s2 P))))
```

Refinement Rollback

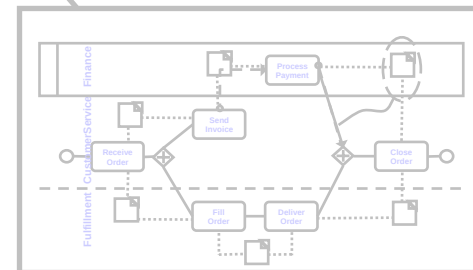
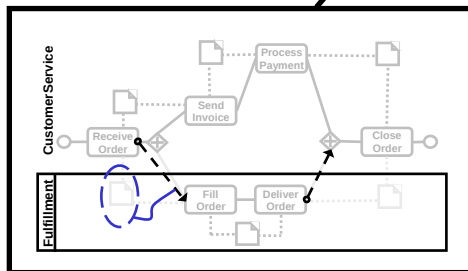
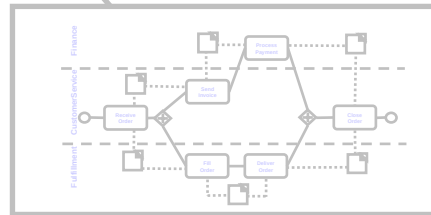
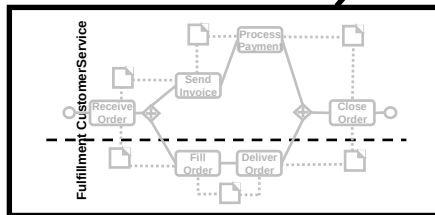
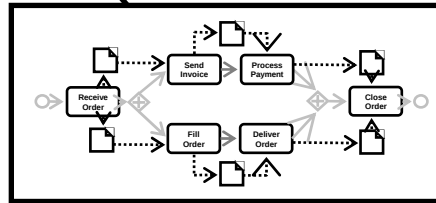
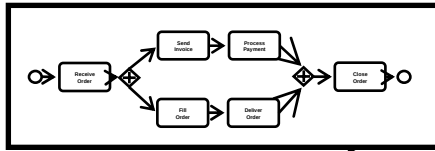
Sequence

Degree
of commitment
(not waterfall)

Input / Output

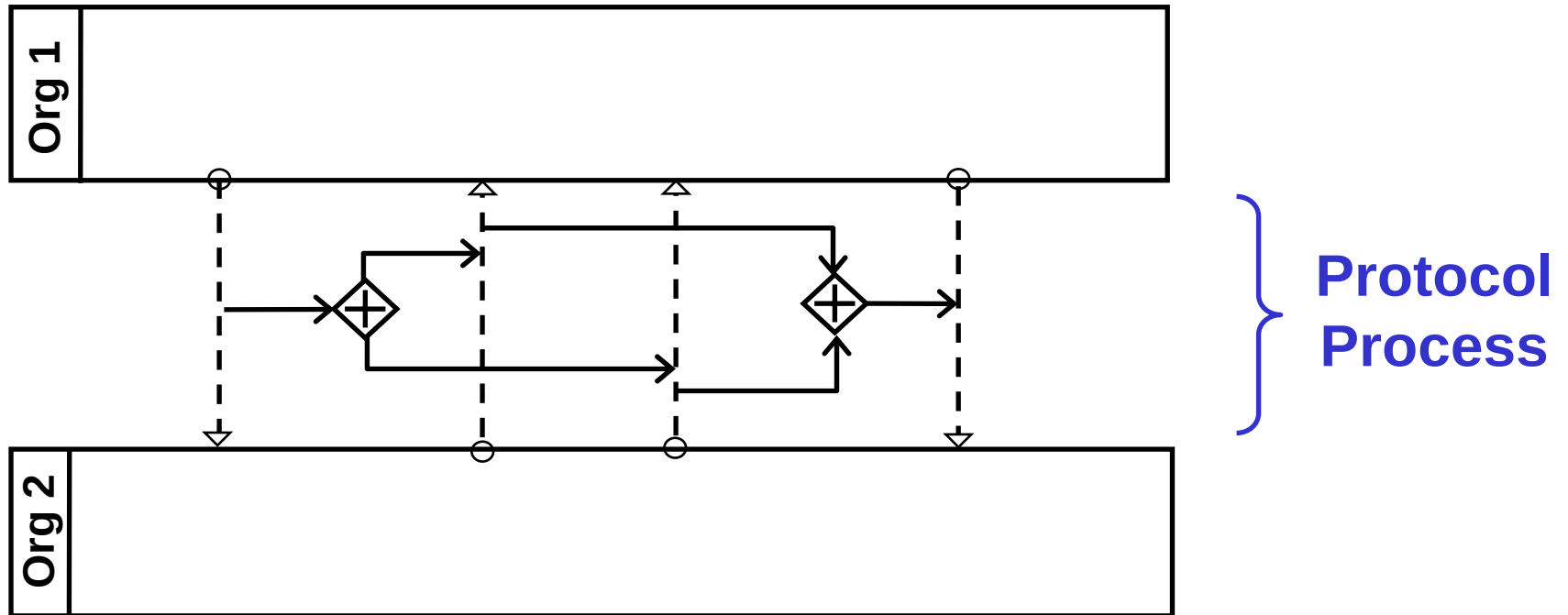
Performers

Coordination



- Alternative process commitments.

Protocol Processes



- **Constraints on messages (as subprocesses)**
 - After first message arrives at Org 2, second two are sent in parallel to Org 1.
 - After those both arrive at Org 1, last message is sent to Org2.
- **For defining standard or contractual interactions (eg, RosettaNet).**

Process / Rule Consistency

- **Business rules and processes are usually represented in incomparable languages.**
- **In PSL, they are both constraints on processes.**
- **Can automatically check consistency of rules and processes.**
 - **By law, a ship heading to a US port has to provide a cargo report to US Customs 24 hours before it sails.**

Process / Rule Consistency

- Customer relationship management processes at IBM too complicated to verify manually.
- Represented company policies as constraints on business processes.
- Tested consistency with PSL translation of those processes.
- Identified ten problems, four of which had not been discovered by rollout.

- Gruninger, M., Atefi, K., and Fox, M.S., "Ontologies to support process integration in enterprise engineering," Computational and Mathematical Organization Theory, 6:381-394, 2000.
- Atefi, K., "Formal models of business process reengineering for design and design validation," Ph.D. Thesis, Enterprise³⁴ Integration Laboratory, Department of Mechanical & Industrial Engineering, University of Toronto, Report TR-EIL-97-1, 1997.

Partial Process Specification

- PSL supports declaring as much or as little as needed about a process.
 - First order constraints on execution model.
- Turns ambiguity (unintentional omission) into abstraction (intentional omission).
 - Did the modeler intend that no other step occur between Paint and Dry?
- Many applications to process: categorization, search and matching, design management, protocols, rule / policy / process integrity.

More Information

- **PSL Specification:**
 - ISO 18269, <http://tinyurl.com/fykru>, 2006.
- **Introduction:**
 - “PSL: A Semantic Domain for Flow Models,” Bock, C., Gruninger, M., Journal of Software and Systems Modeling, 4:2, pp. 209-231, <http://tinyurl.com/8g57x>, May 2005.
- **NIST PSL site:**
 - <http://www.nist.gov/psl>
- **conrad.bock at nist.gov**