The Process Specification Language: Around the World in 80 Axioms

Michael Gruninger
Institute for Systems Research
University of Maryland

Interoperability

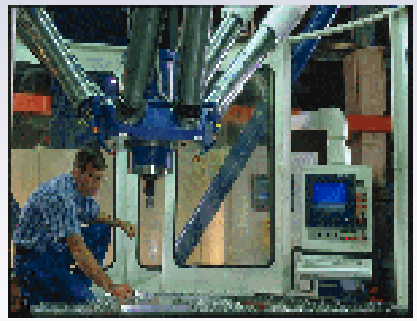
Process Modeler

(ProCAP / KBSI)

Process Planner

(MetCAPP/Agiltech)

Simulator (Quest / Dessault)



Scheduler

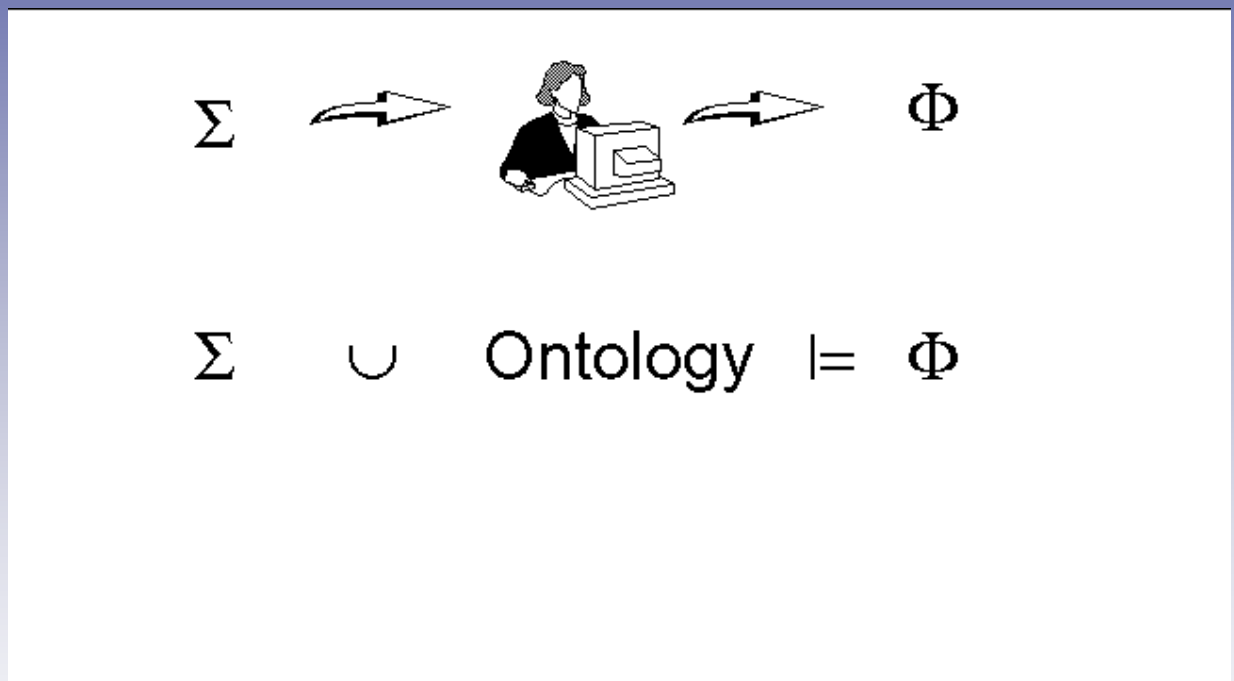
(ILOG Scheduler)

Interoperability Hypothesis

- We are considering interoperability among complete first-order inference engines that exchange first-order sentences.

- *Why first-order logic?*
  - Soundness and completeness guarantees that a sentence is provable from a theory if and only if it is satisfied in all models of the theory.

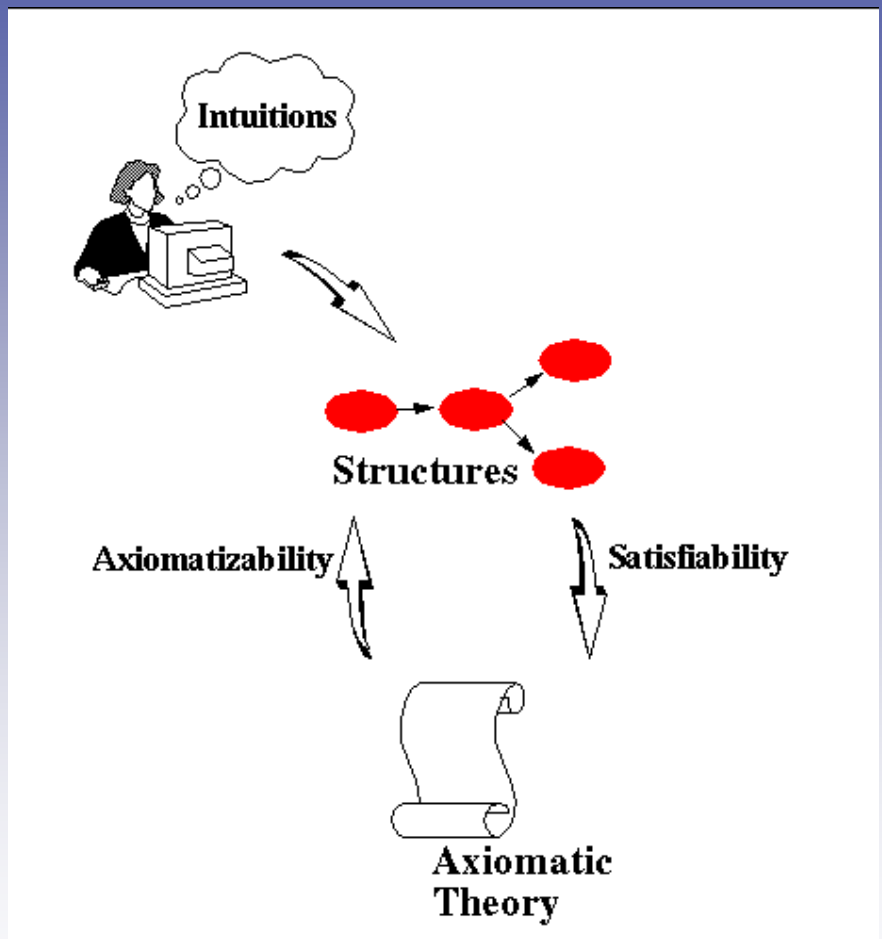
Ontological Stance



Formal Properties of PSL

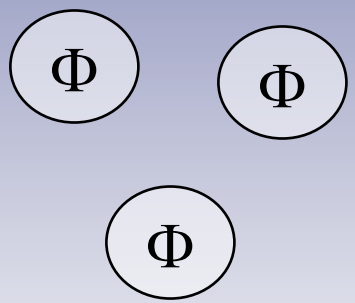
- The meaning of terms in the ontology is characterized by models for first-order logic.
- The PSL Ontology has a first-order axiomatization of the class of models.
- Classes in the ontology arise from classification of the models with respect to invariants (properties of the models preserved by isomorphism).
- Process descriptions are specified by definable types for elements in the models.

Verified Ontologies

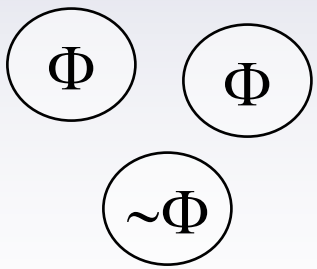


Competency Questions

If we cannot deduce a particular sentence, then there exists a model of the axioms that falsifies the sentence.



$\Gamma \models \Phi$



T does not entail Φ

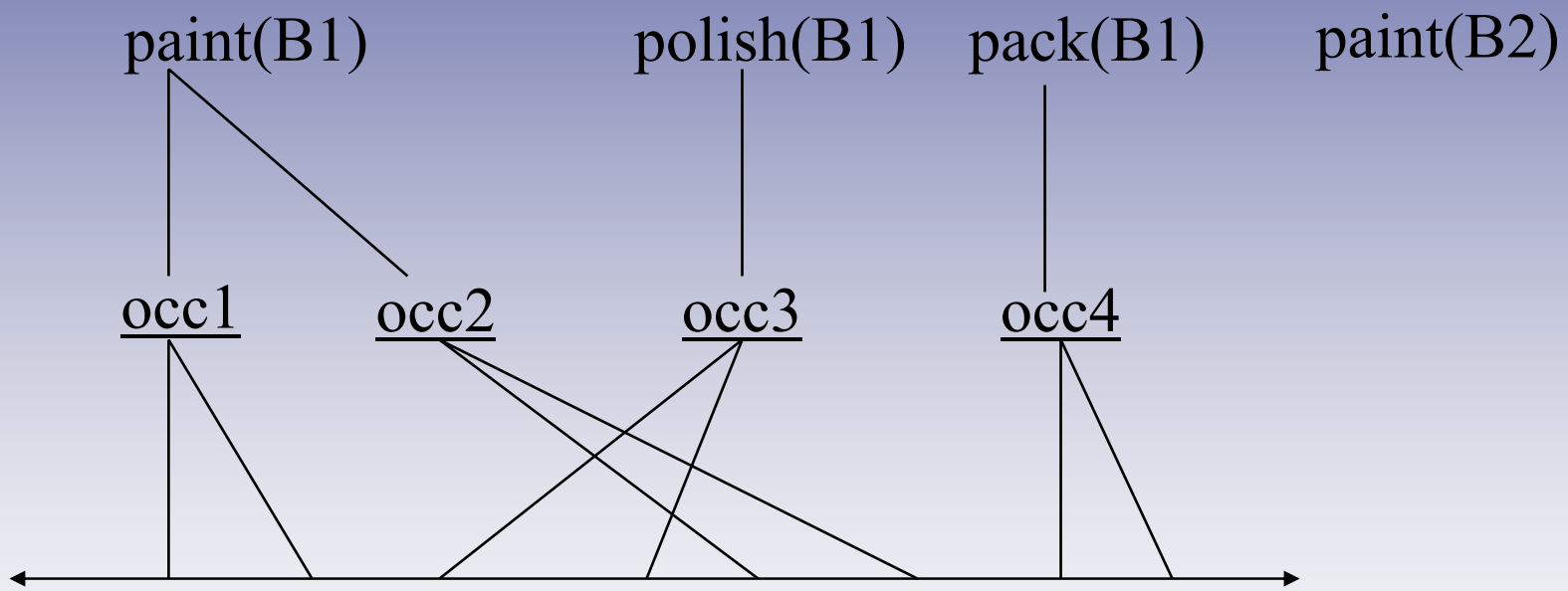
Definability Criterion

- Classes of structures for theories within the PSL Ontology are axiomatized up to elementary equivalence – the theories are satisfied by any model in the class, and any model of the theories is elementarily equivalent to a model in the class. Further, each class of structures is characterized up to isomorphism

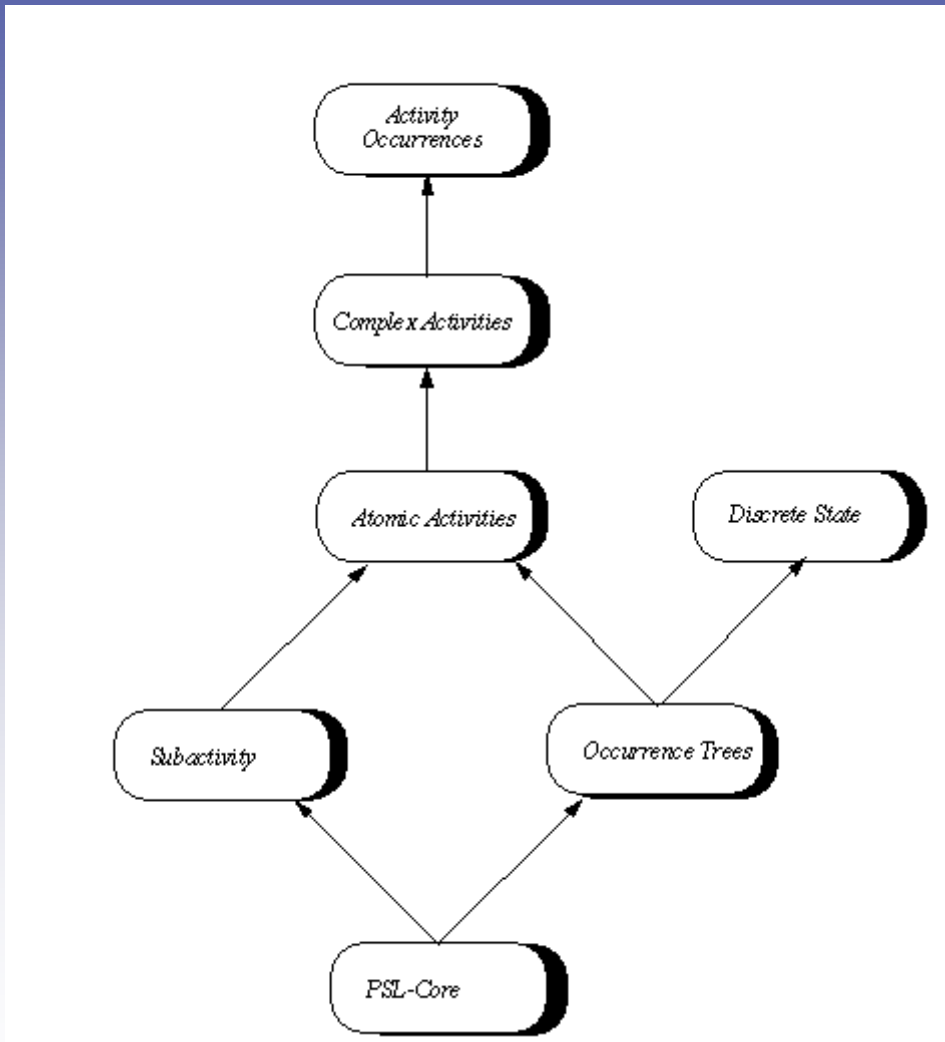
Organization of PSL

- PSL is a modular, extensible ontology capturing concepts required for process specification
- There are currently 300 concepts across 50 extensions of a common core theory (PSL-Core), each with a set of axioms written using the Knowledge Interchange Format.
- Two kinds of extensions:
  - Core theories
  - Defintional extensions

Models of PSL-Core



PSL Core Theories



Additional Core Theories

- Duration
- Subactivity Occurrence Ordering
- Iterated Occurrence Ordering
- Resource Requirements
- Resource Sets
- Activity Performance

Definitional Extensions

- Preserving semantics is equivalent to preserving models of the axioms.
  - *preserving models = isomorphism*
- We classify models by using *invariants* (properties of models that are preserved by isomorphism).
  - automorphism groups, endomorphism semigroups
- Classes of activities and objects are specified using these invariants.

Semantic Translation

Translation definitions specify the mappings between PSL and application ontologies.

Example: The ***AtomicProcess*** in DAML-S maps to the ***activity*** concept in PSL only if the activity is atomic and its preconditions and effects depend only on the state prior to the occurrences of the activity..

```
(forall (?a)
        (iff     (AtomicProcess ?a)
                 (and    (atomic ?a)
                         (markov_precond ?a)
                         (markov_effects ?a))))
```

Twenty Questions

How can we generate translation definitions?

- Each invariant from the classification of models corresponds to a different question.
- Any particular activity or object will have a unique value for the invariant.
- Each possible answer to a question corresponds to a different value for the invariant.

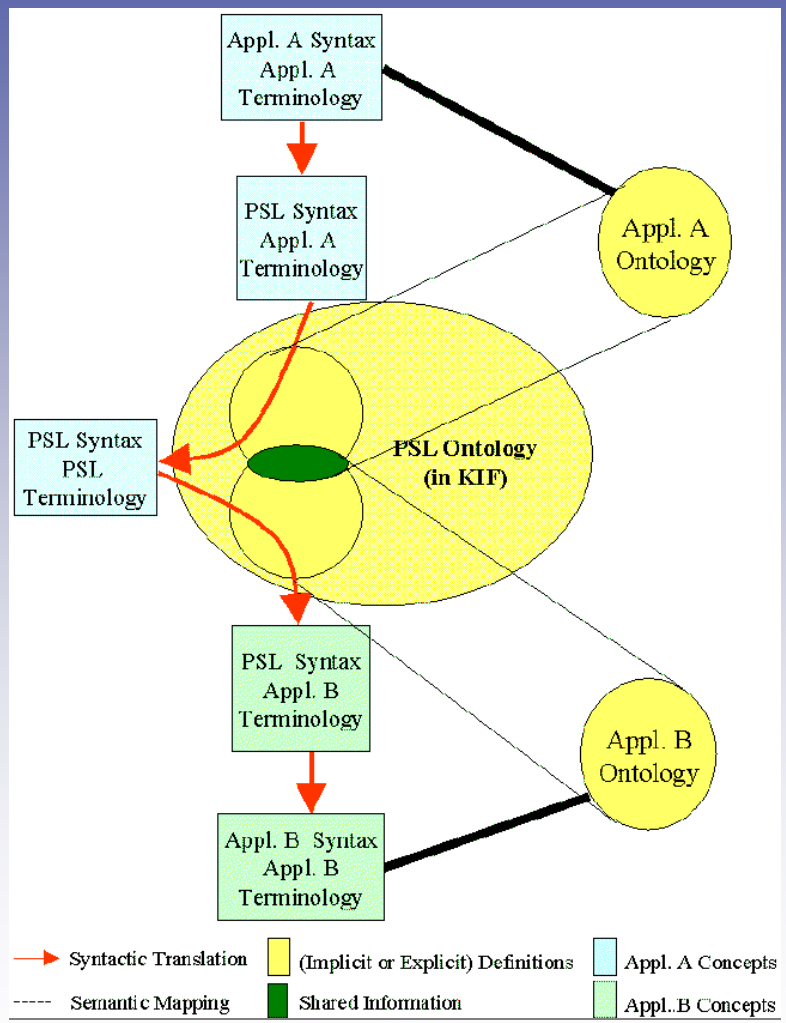
Process Descriptions

- If we shared an ontology of algebraic fields, we would not share arbitrary sentences; rather, we would share polynomials.
- Within PSL, process descriptions are boolean combinations of definable types realized in some model of the ontology.
- Example: precondition axioms are types for markov_precond activities

Demonstrations

- Automatic analysis of software application interoperability from semantic mappings.
- Automated analysis of business processes
- Self-coordinating software agents based on published process specifications.
- Construction project management
- Behaviour recognition

Semantic Integration



- Issue:
  - Automatic analysis of software application interoperability from semantic mappings.
- Problem:
  - Automatically determine which concepts are shared by two software applications.
- Solution:
  - Twenty Questions Tool semi-automatically generates mappings between PSL and application ontologies.
  - Use automated reasoners to compare semantic mappings for different applications.

Gruninger, M. and Kopena, J. (2003) Semantic Integration through Invariants, to appear in AI Magazine.

Business Process Analysis

- Issue:
  - Enhance computer support for enterprise design, integration, and decision-making
- Problem:
  - Customer Relationship Management processes at IBM Canada too complicated to verify manually.
- Solution:
  - Used a Prolog implementation of PSL to represent the processes and determine consistency of policies. Identified ten problems, four of which had not been discovered even at the time of rollout

Gruninger, M., Atefi, K., and Fox, M.S., (2000) Ontologies to support process integration in enterprise engineering, Computational and Mathematical Organization Theory, 6:381-394.

Coordinating Software Agents

- Issue:
  - Handle very complex and diverse systems.
- Problem:
  - Mobile software agents on an ad hoc wireless network must locate each other to integrate sensor data for target identification.
- Solution:
  - Specify the communication capabilities of the agents as activities within PSL
  - Locate potential collaborators by deduction (using Otter)

Kopena, J. (2004) Service Descriptions, Matchmaking, Composition: Reasoning on Actions and Beliefs, Technical Report, Geometric and Intelligent Computing Laboratory, Drexel University.

Construction Project Management

- Issue:
  - Coordinate knowledge existing across organizations and disciplines
- Problem:
  - Modify project schedules based on updated information from contractors and subcontractors.
- Solution:
  - Contractors use PDAs to send updates to the project server; PSL is used to determine if these updates are inconsistent with the project schedule (using Otter), and then communicate the changes to other contractors.

Cheng, J., Gruninger, M., Sriram, R., and Law, K. (2003) Process Specification Language for project scheduling information exchange, International Journal of IT in Architecture, Engineering and Construction, 1:307-328.

Behavior Recognition

- Scenario:
  - Predict future behaviors of other autonomous vehicles by observing their current behavior, to avoid accidents or improve performance.
- Potential solution:
  - Specify the driving behaviors as complex activities in PSL
  - Constraint satisfaction techniques (using Theorist) determine which driving behaviors are consistent with the observations.

Major Project Milestones

- April 2000: PSL accepted as a New Work Item ISO 18629 within ISO SC4/SC5
- October 2001: ISO 18629-1 passed CD ballot
- June 2002: ISO 18629-12 (Outer Core) submitted for CD ballot.
- September 2002: PSL 2.0 released (including grammars for process descriptions)
- November 2002: ISO 18629-11 (PSL-Core) passed CD ballot
- February 2004: ISO 18629-1 passed DIS ballot

Future Directions

- Formal characterization of the consistency and completeness of all remaining PSL extensions.
- Implementation of Twenty Questions Semantic Mapping tool
- Specification of Process Information Exchange protocols to support self-integrating systems.

Further Questions?

Michael Gruninger
gruning@nist.gov
(301) 975-6536

http://www.nist.gov/psl