# *The Role of Ontologies for Agent-oriented Systems*

**Line Pouchard**

Data Science Group

Computer Science and Mathematics Division

Oak Ridge National Laboratory

# *Outline*

1. Definitions and use of ontologies
2. Agent-oriented systems
3. Using ontologies in agent-oriented systems
4. The Process Specification Language
5. Process exchange using PSL
6. Summary

# *1.1 What are (domain) ontologies?*

- The basic terms and relations comprising the vocabulary of a topic area
- A set of definitions for these terms
- The rules for combining terms and relations
- Provide a theory for the domain

# *1.2. Some definitions*

- Definitions [in an ontology] associate the names of entities of discourse (i.e. classes, relations, functions or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these names. (Huhns and Singh, 1997, “Ontologies for agents”)
- A domain theory that specify domain-specific vocabulary of entities, classes, properties, predicates, functions and a set of relationships that necessarily hold among these vocabulary items (Fikes & Farquhar, 1999)

# *1.2. Why are ontologies useful?*

- Provide definitions and axioms constraining the use of terms that are readable by machines and understandable by humans
- Allow hierarchical classification systems, generalization and inheritance, aggregation, and a greater variety of structural relations than taxonomies and controlled vocabularies

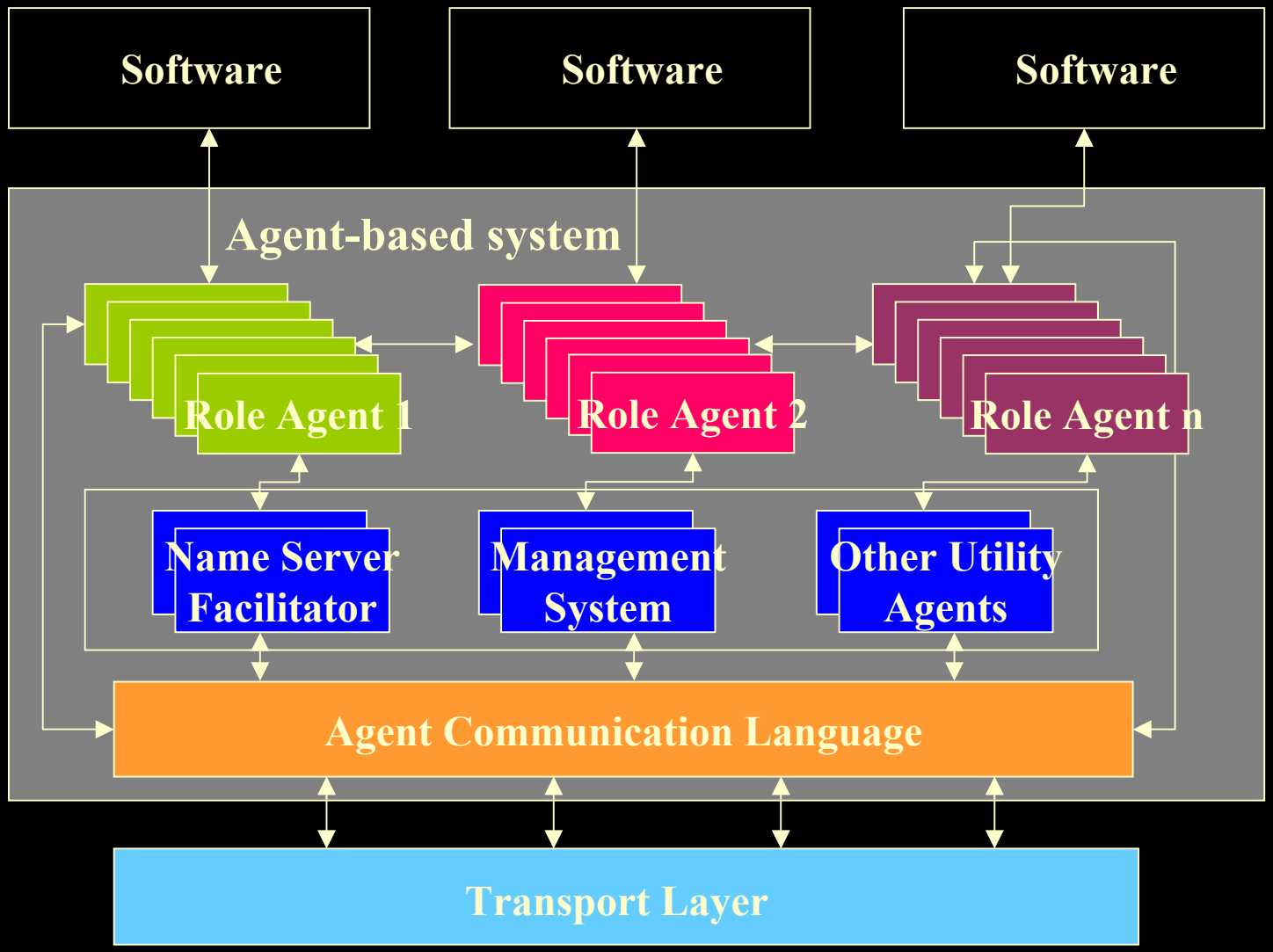
# *1.3. When should ontologies be used?*

- by humans: for providing a common frame of reference and some consensus on entities in a domain
- by machines:
  - data schemas
  - system inter-operability based on semantics
  - agent-based systems

# *2. Features of software agents*

- Programs acting upon changes in their environment (can be automatically queried)
- Encodes a set of rules for individual behavior
- Provide a powerful abstraction for intuitively representing user and component interactions with the system
- May rely on high-level message interaction
- Autonomous, goal-directed, adaptive (deliberative or reactive, learning), collaborative, mobile

# 2.2. A FIPA Agent Platform



# *2.3. Agent-oriented Systems*

- A system where autonomous agents must coordinate their actions:
  - interconnecting agents are created on different platforms
  - agents interact for a purpose, form a team
  - provide capabilities beyond those of individual agents (concurrency, common interface, proper workflow between tasks, access to heterogeneous and distributed resources)

# *2.4. Interaction between agents*

- cooperative: each agent's goal is subservient to the over-arching goal of the system
- collaborative: each agent pursues its own goals, and does not recognize a higher goal (trading agents in e-commerce)
- both cooperative and collaborative models need coordination (task negotiation and conflict resolution)

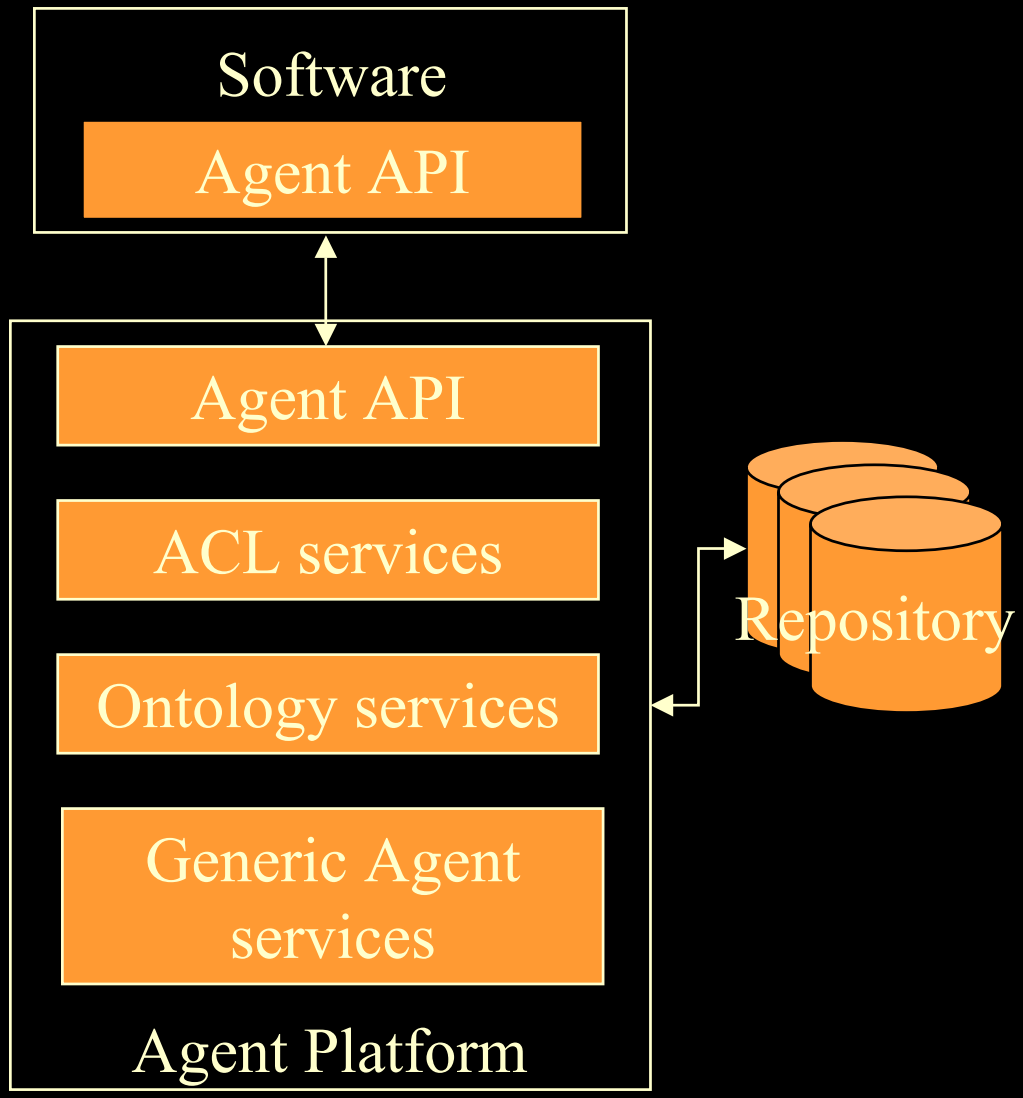
# *2.5 Other important characteristics*

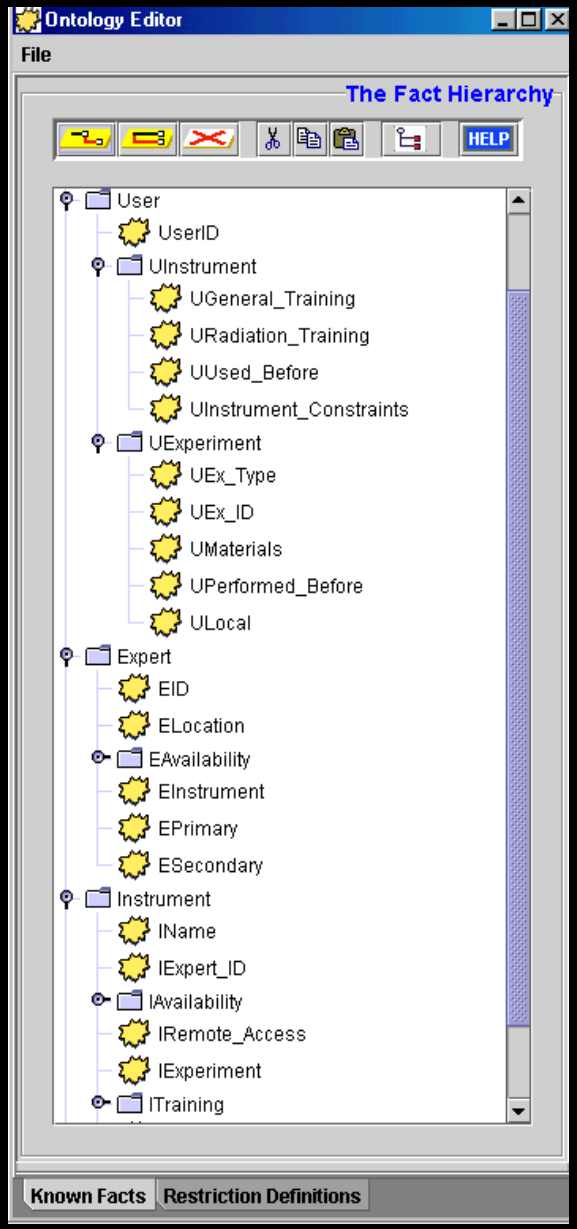
- The degree of interaction
  - number of agents
  - degree of concurrency
- The model of control
  - centralized control: a utility agent regulates all actions
  - distributed control: the behavior of the system emerges from the agents' interaction

# *3. Ontologies in agent-based systems*

- Agent-based systems may use an Agent Communication Language (ACL) for interactions between agents
- Heterogeneous agents need to agree on a common data model that include semantics: the ontology is a common reference for a robust representation of concepts
- Ontologies may be encoded in several languages:
  - KIF, XML, RDF schemas, DAML+OIL

# 3.2.Ontology layer



# *3.3 Agent Communication Language (ACL)*

- Use of performatives (one per message):
  - ask-if, tell, subscribe, reply, recommend-all
  - inform-if, confirm, request, propose
- Each performative has attribute/value pairs:
  - (:content, :language, :sender, :receiver, :ontology …)

# *3.4. Ontology attribute in an ACL performative*

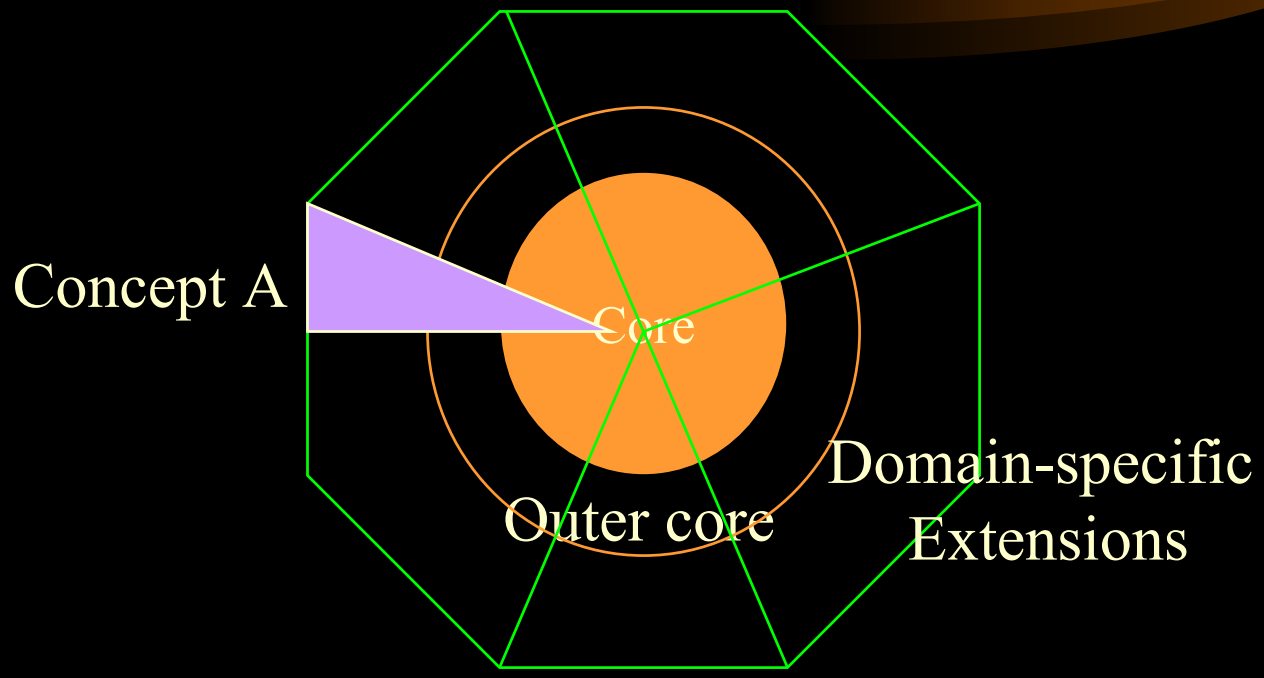
- (inform
  :sender (ORNLcollaborator3)
  :receiver (PNNLcollaborator2)
  :in-reply-to (PNNLcollaborator2 _local_dstruct100)
  :ontology (climate_model_34)
  :content (temperature@surface PNNLcollaborator2_goal
  187 (:type :id :modifier :attributes (Fahrenheit
  65.5)
  )
  )

# *4. Process Specification Language*

- A neutral, standard description of manufacturing processes expressed in KIF
- A core ontology and extensions for each process domain
- Translation mechanisms between individual applications and PSL
- One pilot implementation completed

http://www.mel.nist.gov/psl

# 4.2. Process Specification Language



# *4.3. Some axioms in PSL*

- Core concepts
  - activity, object, timepoint
  - activity occurrence
- Concepts for object
  - resource roles, paths, sets, usage, etc...
  - states
- Relations
  - before, begin_of, participates_in

http://www.mel.nist.gov/psl/psl-ontology/

Everything is either an activity, object, or a timepoint.

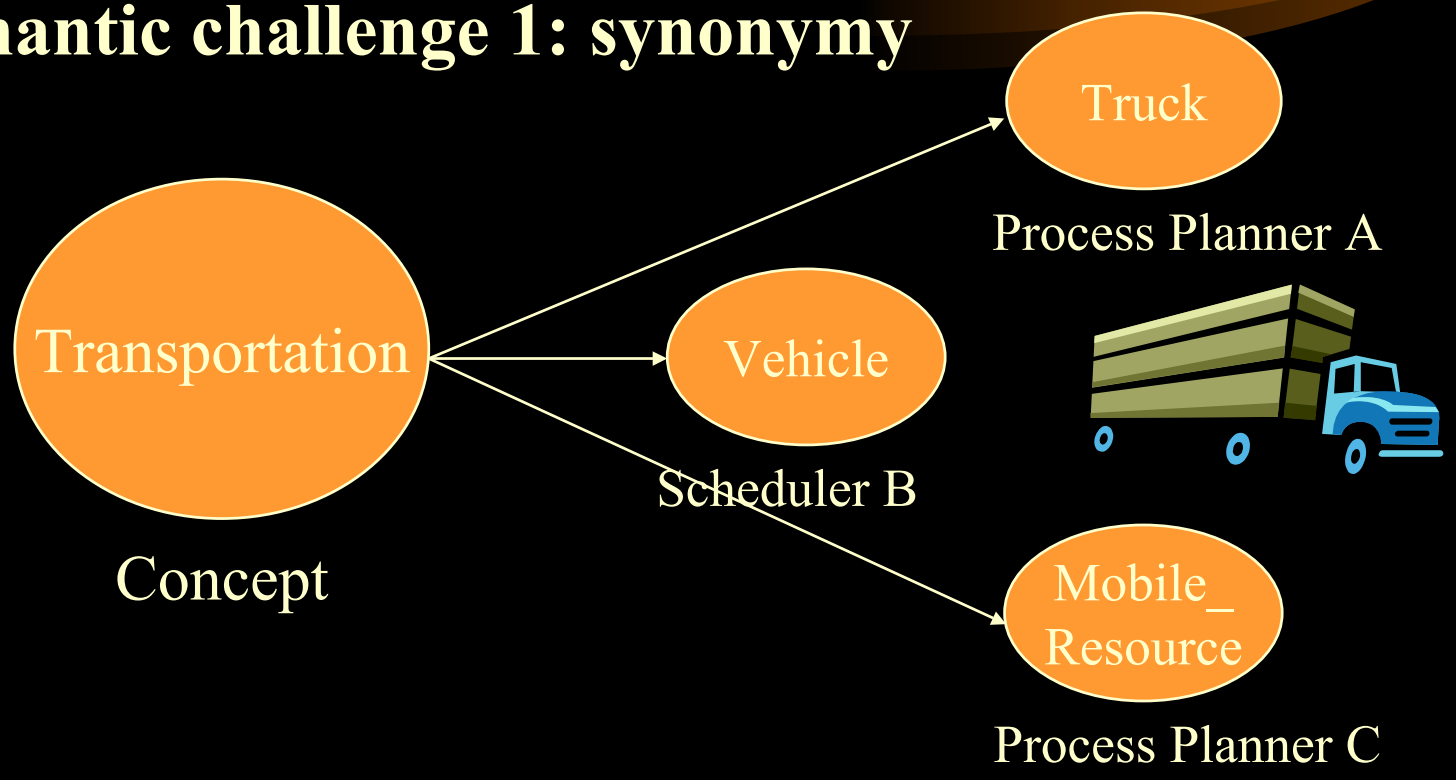
```
(forall (?x)
        (or    (activity ?x)
               (activity-occurrence ?x)
               (object ?x)
               (timepoint ?x)))
```

For every interval over which an activity occurs, there exists an occurrence of the activity.

```
(forall (?a)
     (<=>   (exists (?occ)
                (occurrence ?occ ?a))
            (exists (?int)
                (occurs_over ?a ?int))))
```

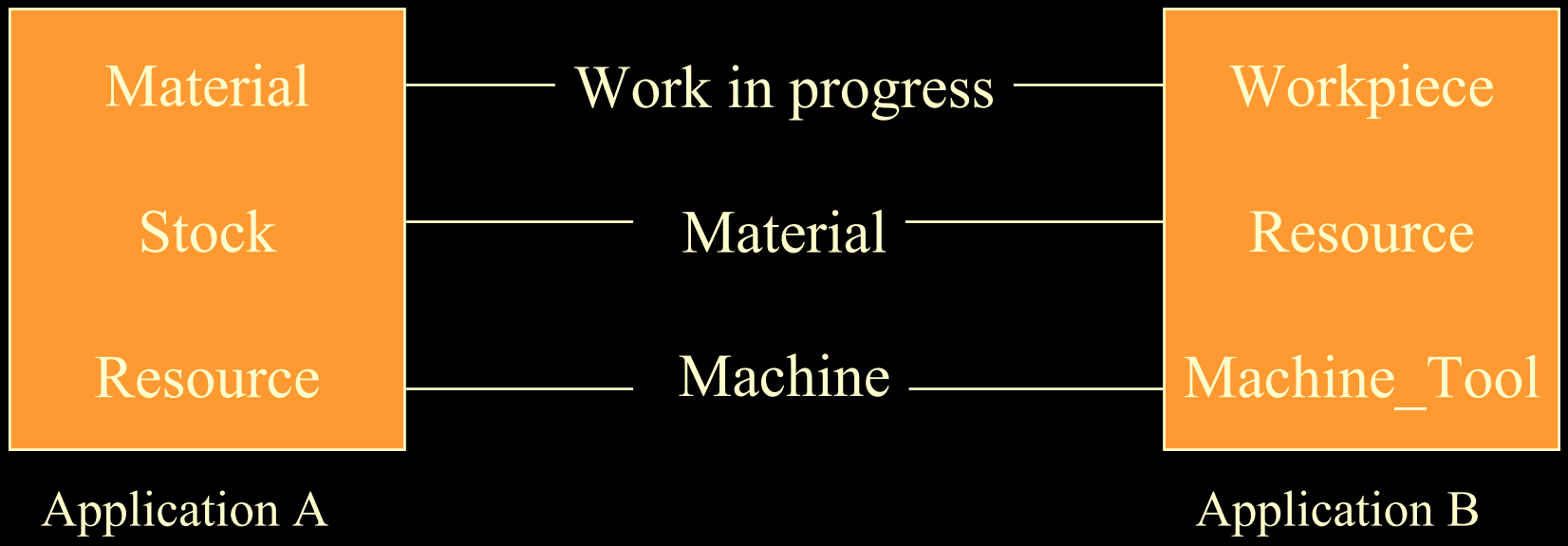
# *4.4. Challenges to inter-operability*

## Semantic challenge 1: synonymy

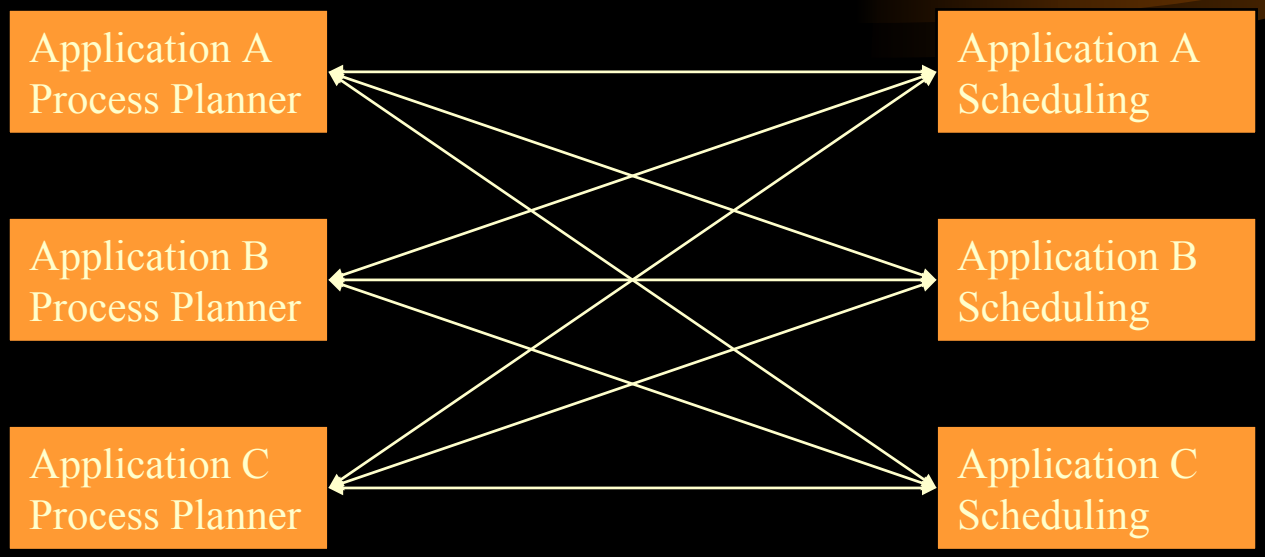


# *4.5. Challenges to inter-operability*

**Semantic challenge 2: term ambiguity**

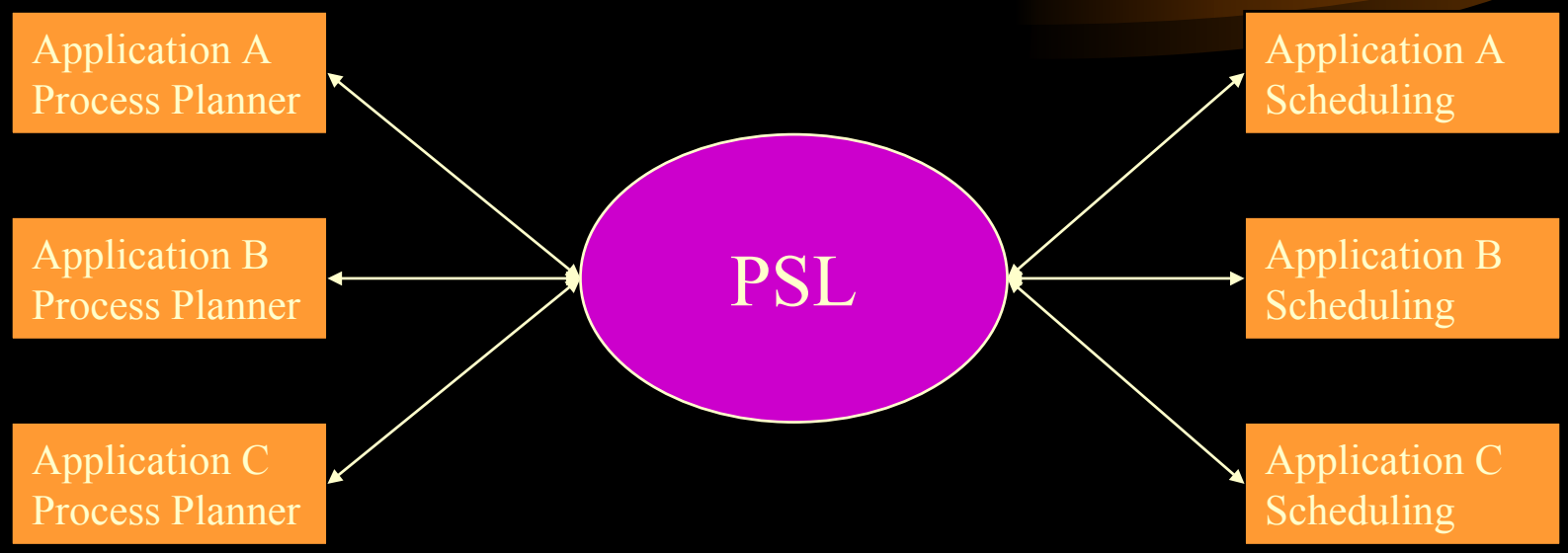


# *4.6 . Scenario 1 point-to-point translation*



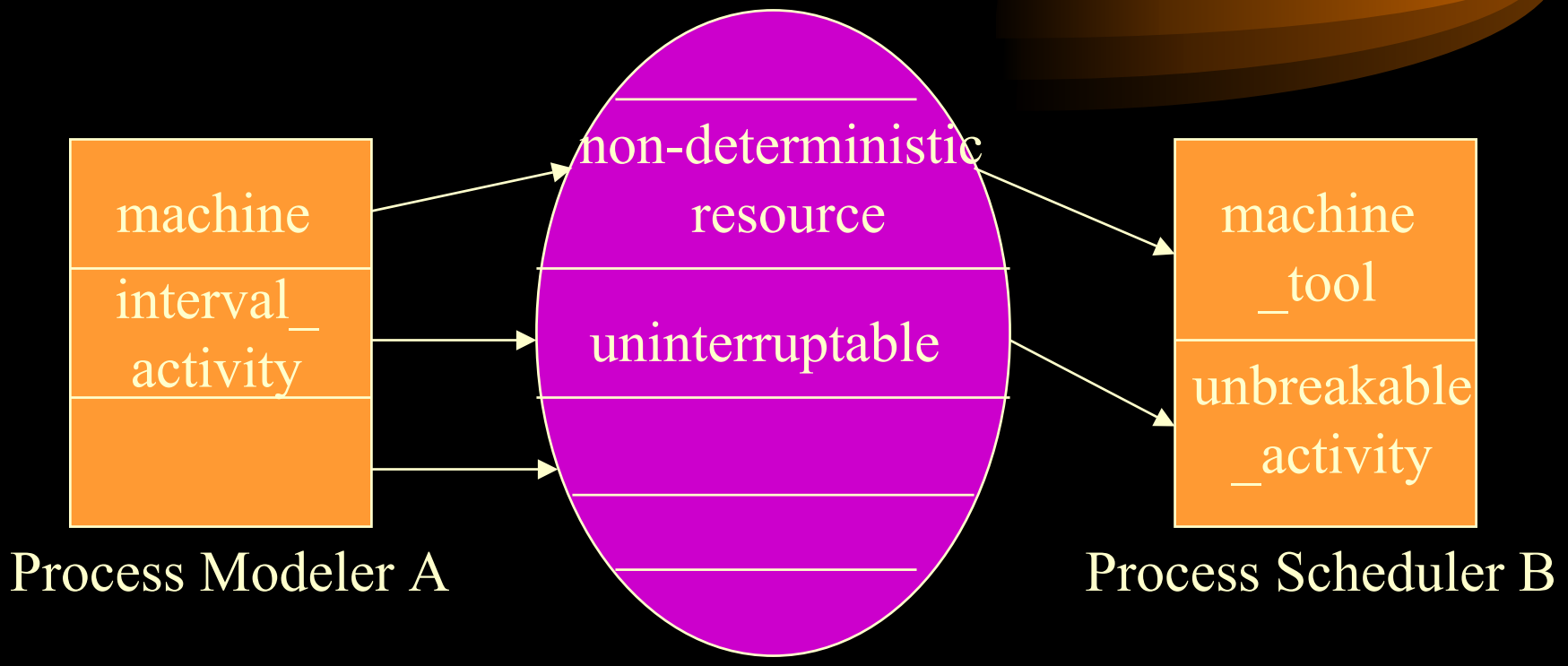

n planners, m schedulers ⟶ n*m translations

# *4.7. Scenario 2 translation using PSL*




n planners, m schedulers ⟶ n + m translations

# *5.1. Process exchange*




PSL concepts used in the exchange

# *5.2. Process Exchange using PSL*

- The ontology for each application is expressed using PSL concepts
- A direct (unconditional) mapping can occur
- OR the application's term is more restrictive =>constraints on its use
- OR PSL is extended to accommodate a new concept

# 5.3. Expressing Application A concepts using PSL

1

```
(forall (?r)
    (=> (inject_mold)
        (rA ?r)))
```

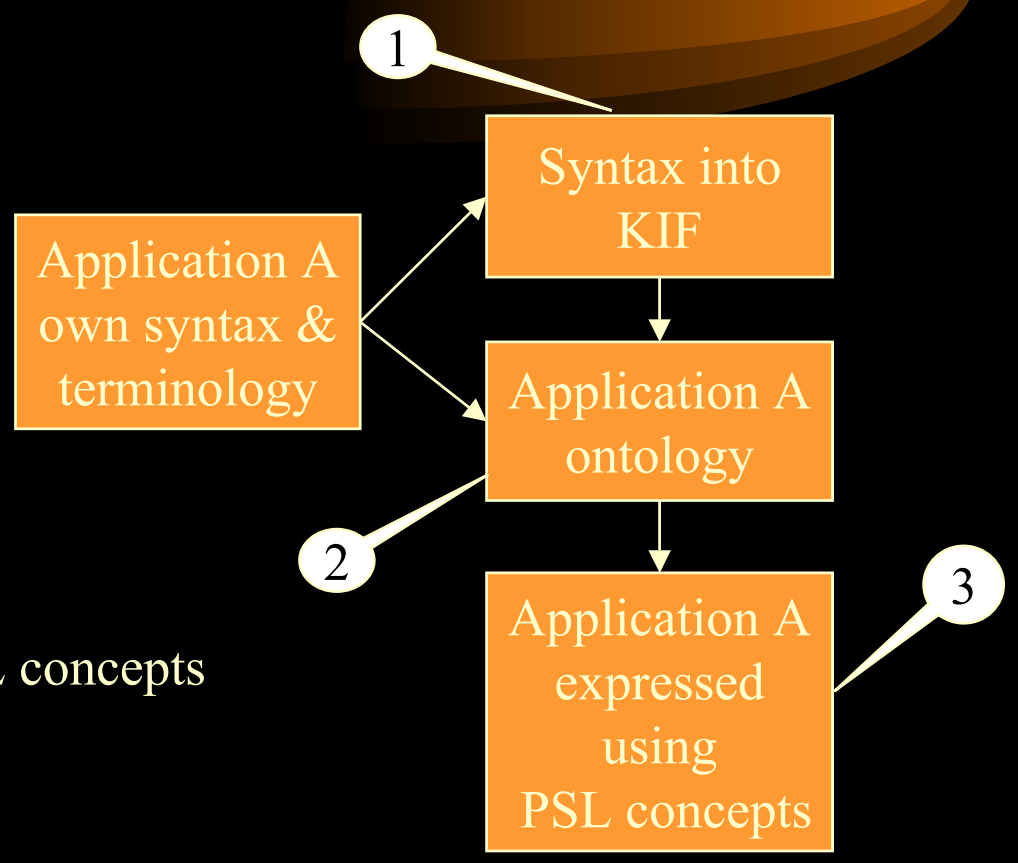
2

```
(forall (?r ?a)
    (<=> (rA ?r)
        (exists (?a)
        (reusable ?r ?a))))
```

3

Unconditional mapping to PSL concepts

```
(forall (?r)
   (<=> (rA ?r)
        (and resource ?r)
        (reusable ?)))
```

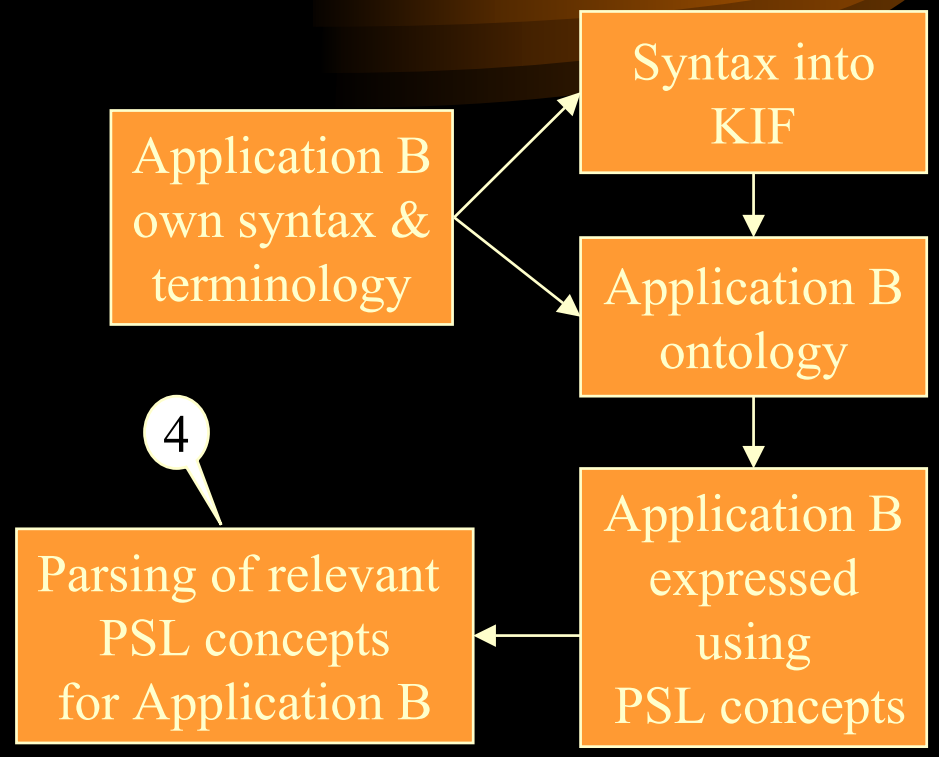


# 5.4. Expressing Application B concepts using PSL

4

One additional step: inverting the table

(Application B => PSL)
(PSL => Application B)



# *5.5 ISO Standardization Effort*

- ISO 18629-Parts 1-x: Industrial Automation Systems and Integration - PSL -
- ISO 18629- Part 1 (Overview and Basic Principles) was accepted by CD ballot and is moving to Draft International Standard level in ISO TC184, joint sub-committee JW8, SC4-SC5, in 2000

# *6. Summary*

- Ontologies in agent-based systems provide a semantic model for application content. They can be called as an attribute of the ACL.
- PSL as a domain ontology for manufacturing concepts
- PSL as an interchange language towards resolving semantic challenges for inter-operability

# *Government Disclaimer*

- The submitted manuscript has been authored by a contractor of the U.S. Government under Contract No. DE-AC05-00OR22725. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.