

Ontology-based representation of compliance requirements for service processes

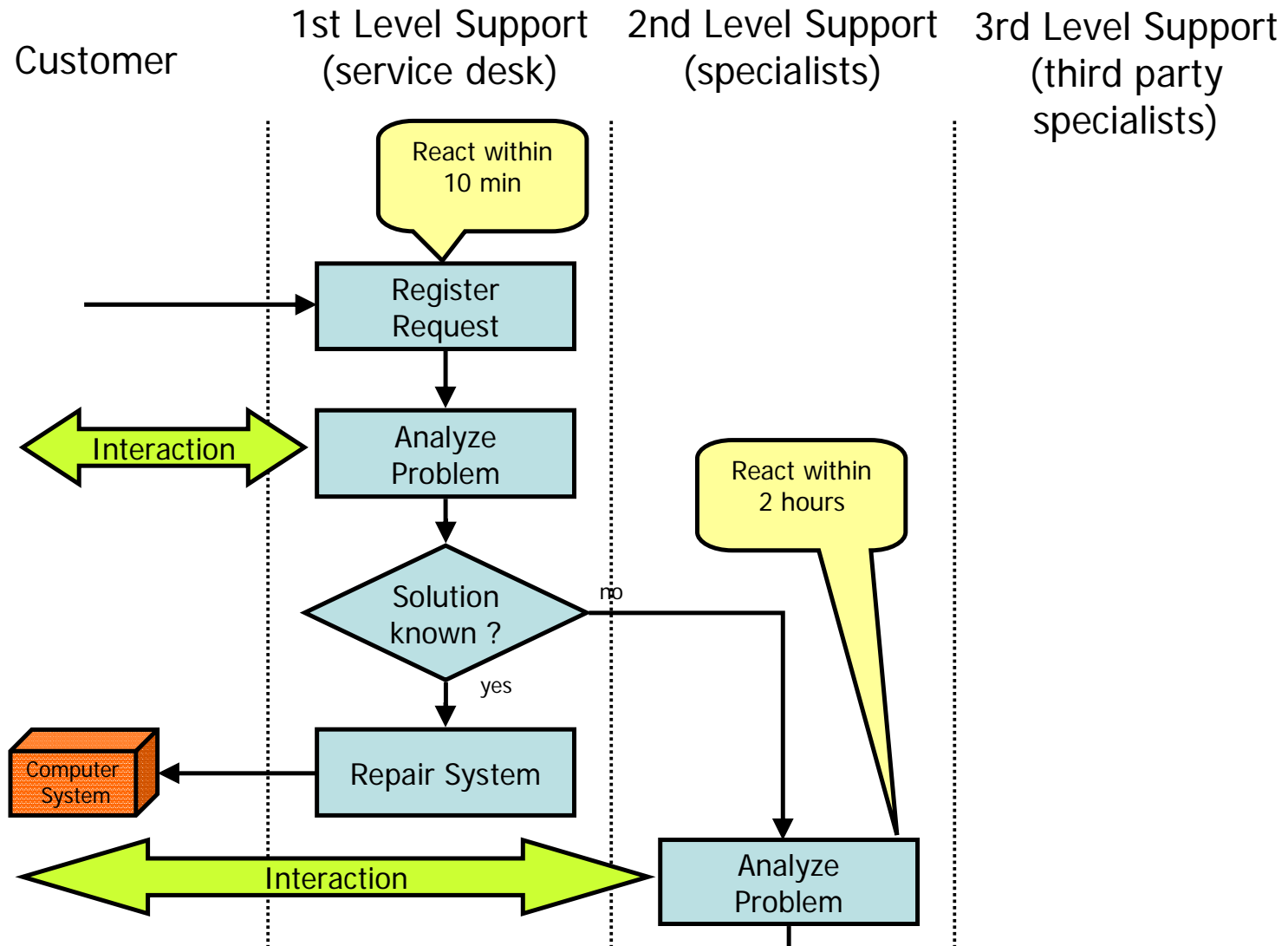
Rainer Schmidt, Roy Oberhauser
Aalen University, Germany

Christian Bartsch
Information Process Engineering (IPE)
Research Center for Information Technologies (FZI)
Karlsruhe, Germany

Overview

- Context and Background
 - What are service processes?
 - Why are they different?
- Problem and Motivation
- Solution approach
- Summary

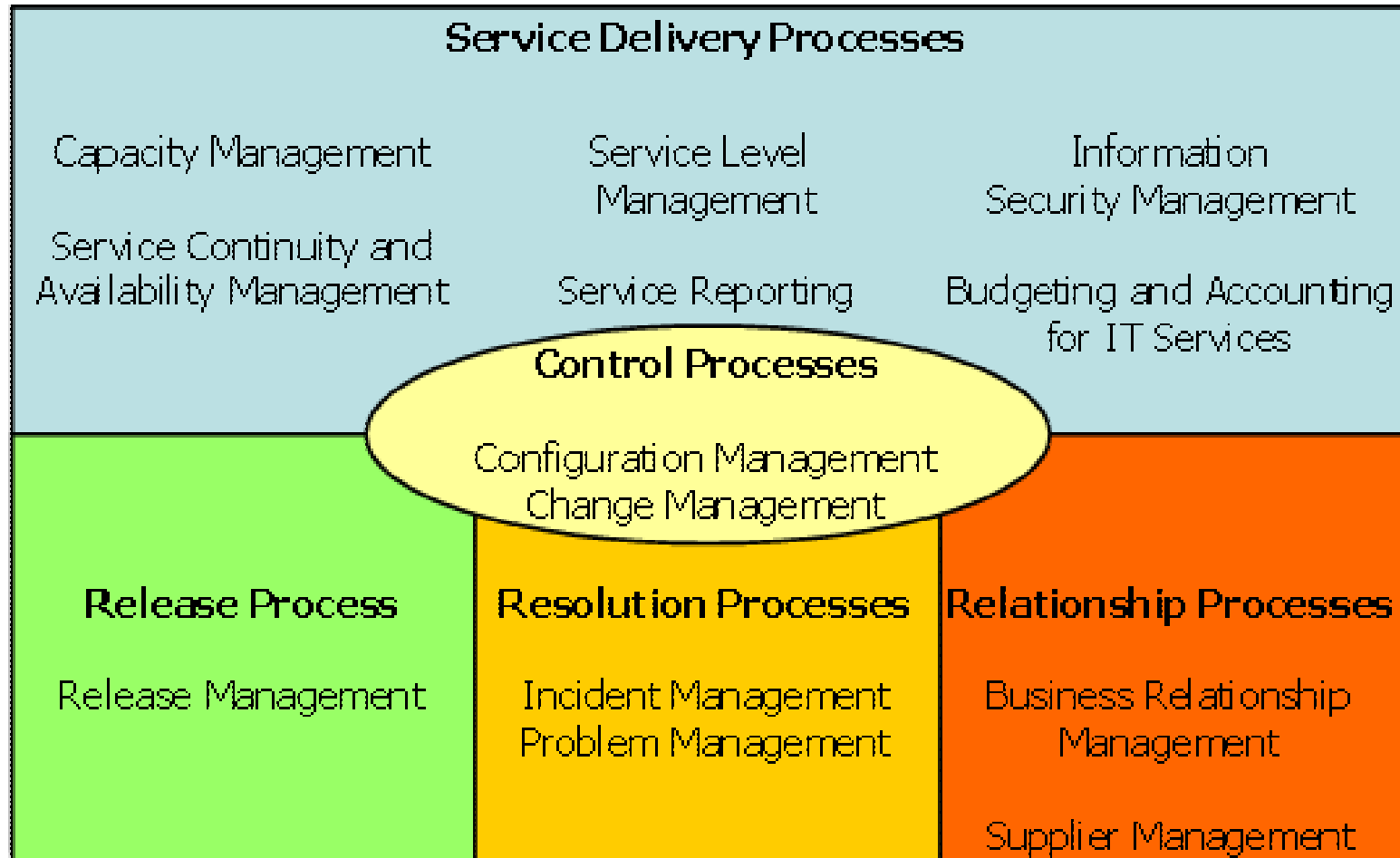
What are service processes? Example: Service Desk



Service Processes

- Are executed to provide services
 - E.g., repair a computer system
- Quality cannot be measured beforehand
- Convince the customer that the service provided will have the quality specified
 - Customer cares about process, not just outcome
- Quality standards (ISO 20000) define compulsory elements and structures in these processes
 - Assure a certain level of quality
- Service processes are certified for compliance
 - E. g., with ISO 20000 to convince the customer of the quality of services

ISO 20000



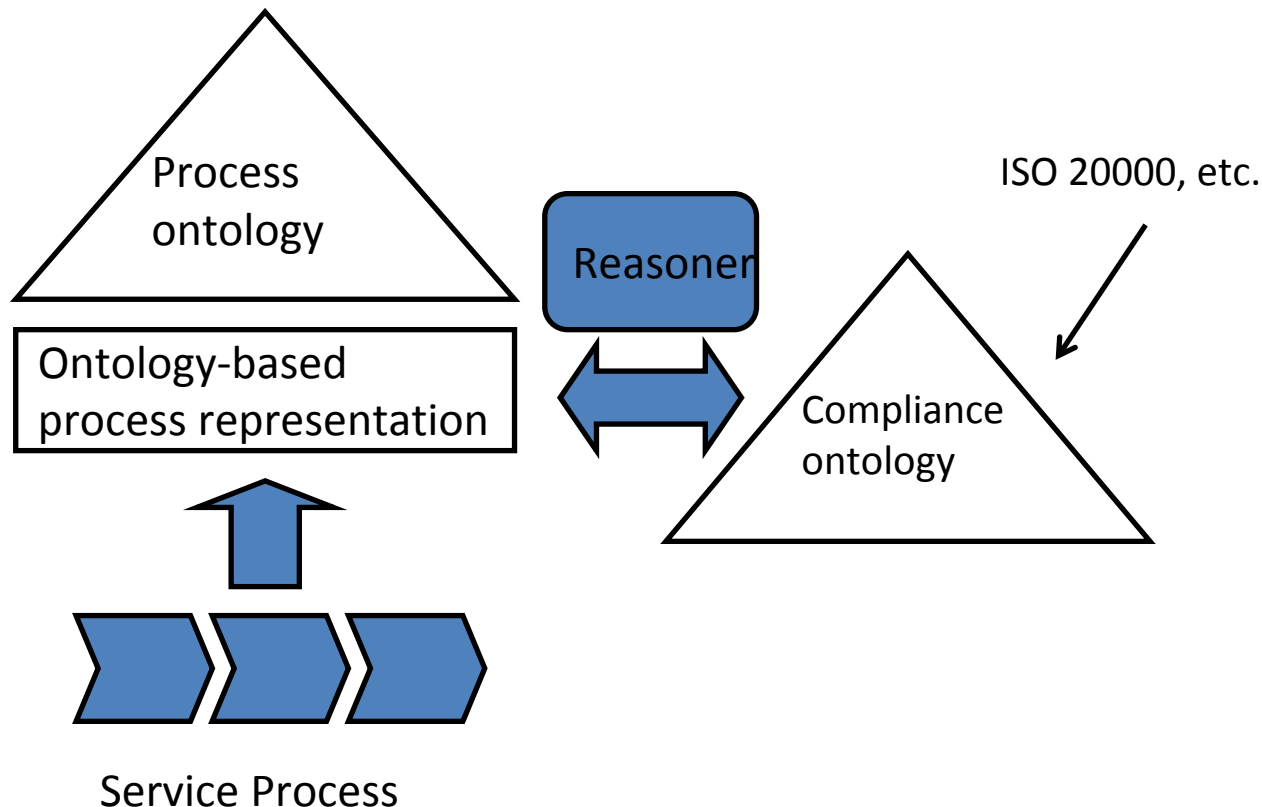
Source: ISO 20000

Problem and Motivation

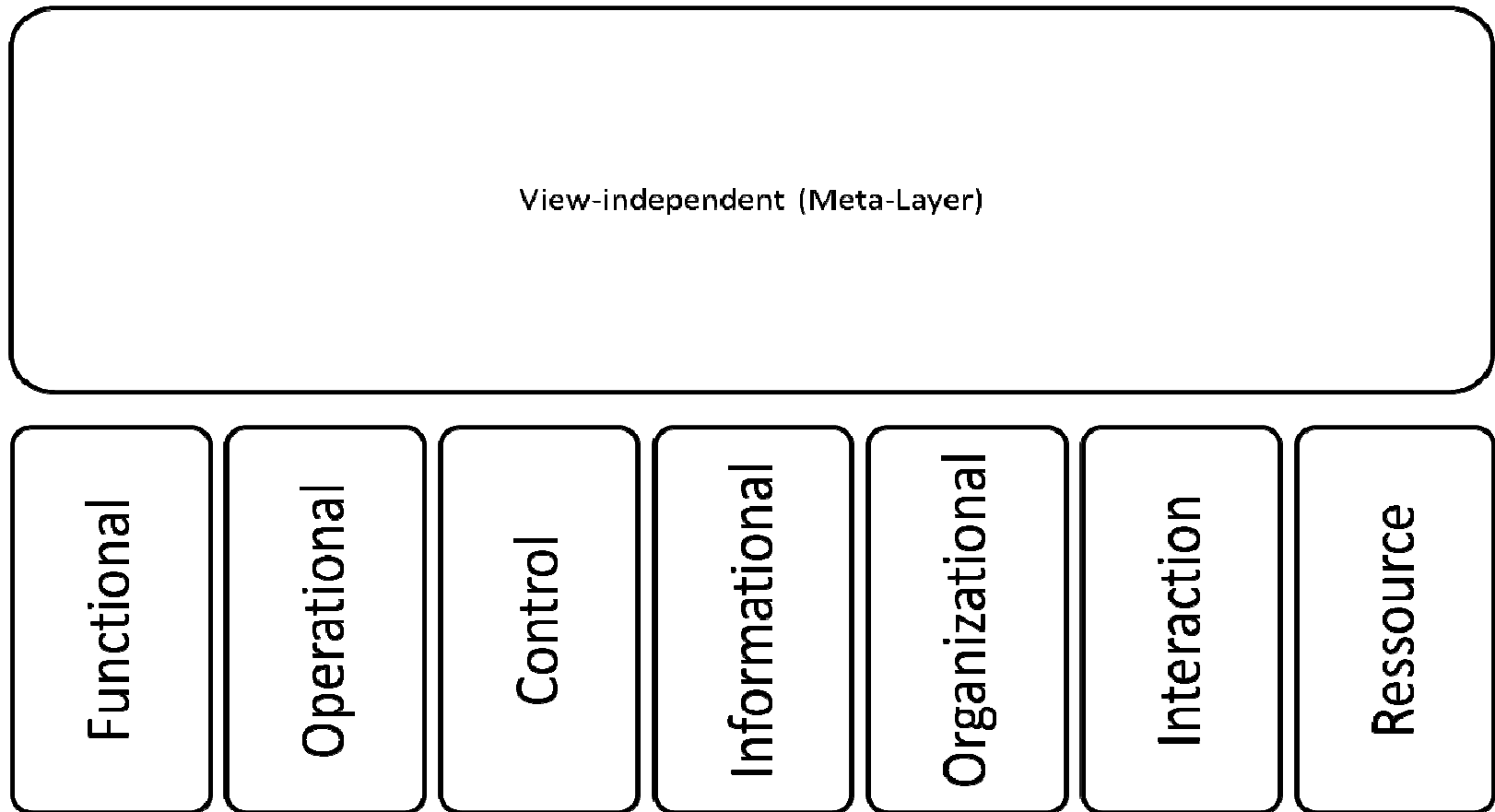
- Quality standards have to be very abstract to allow different implementations
- However, the large semantic gap between quality standards and executable processes requires enormous effort to test for compliance
- Changes to service processes (which happen frequently) require that the compliance of the change has to be verified

Solution Approach

- Represent service processes and quality standards as ontologies
- Reduce compliance checking effort via reasoners

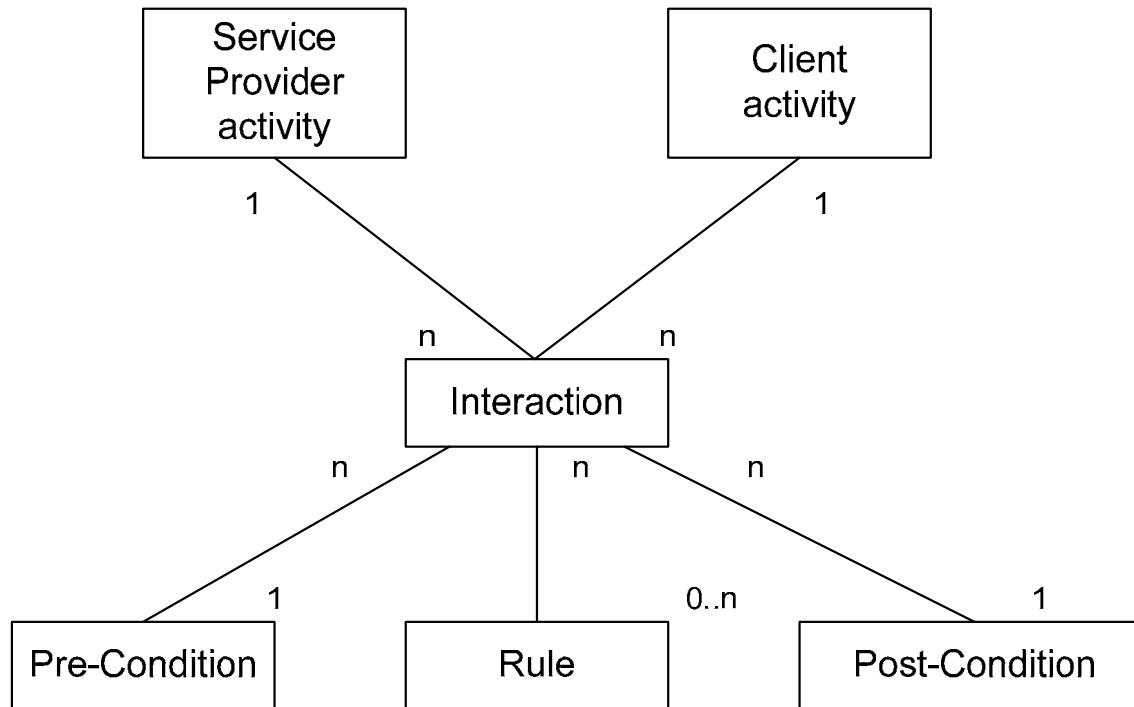


View-Oriented Structure of the Ontology

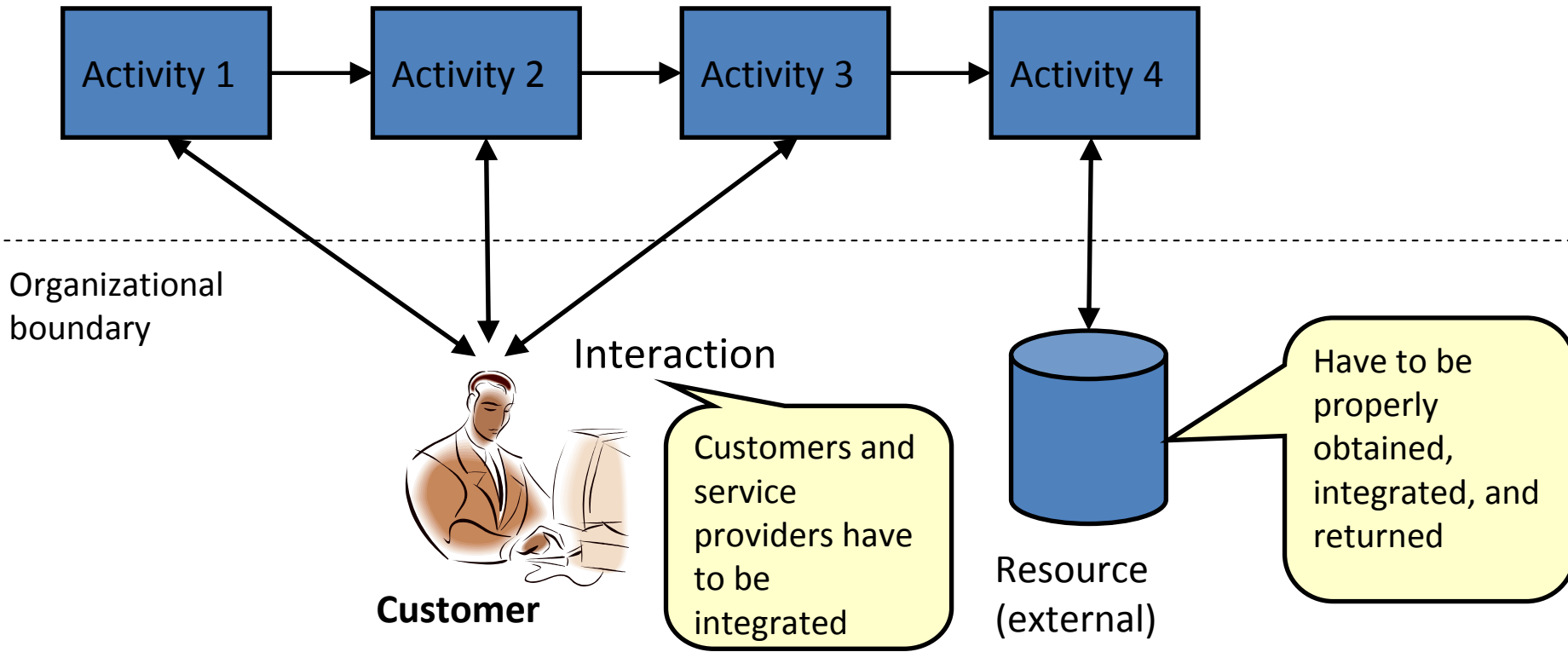


Interactions

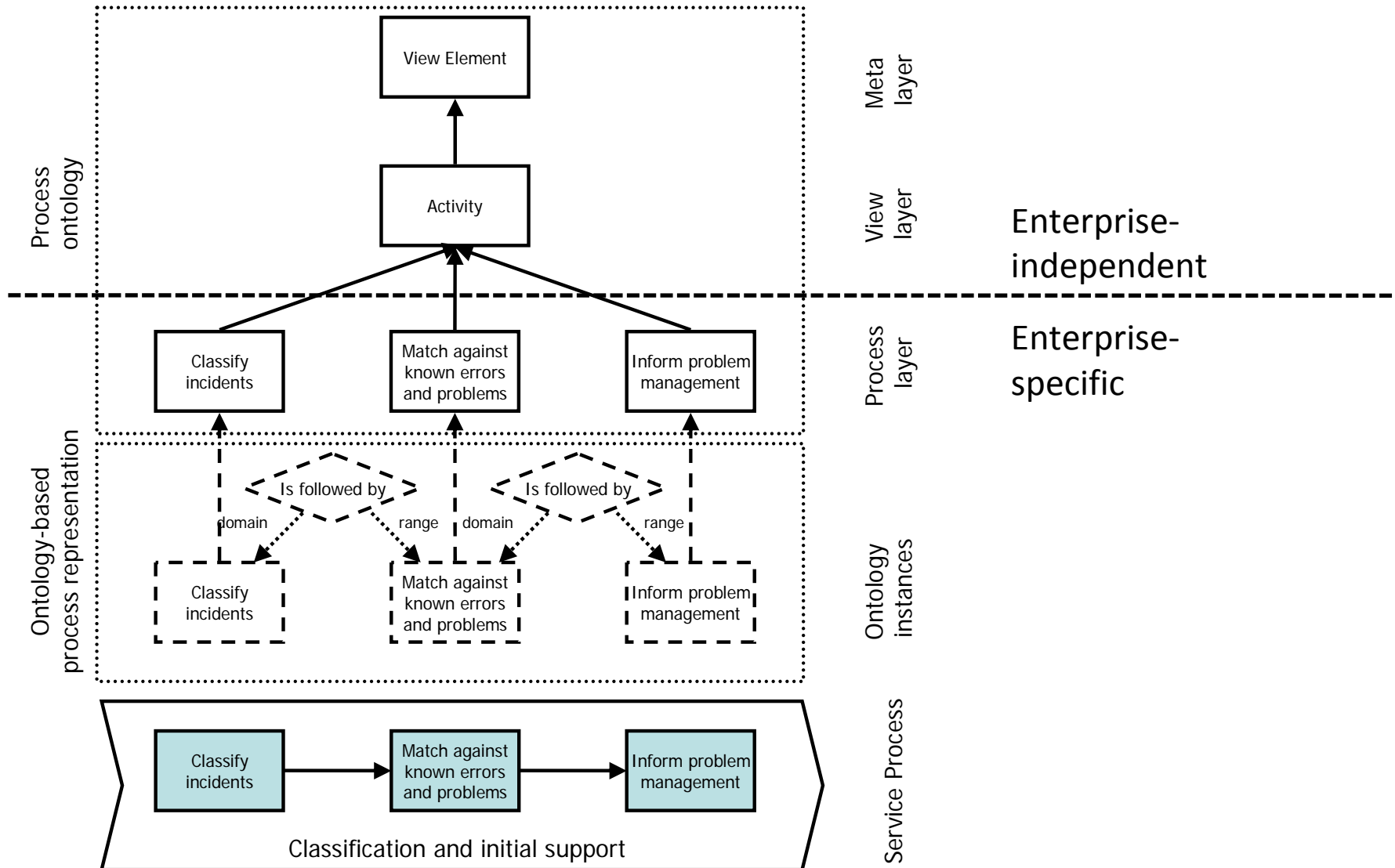
- Follow pre-defined patterns
- Are documented (e.g., for disputes)
- Contain multitude of communication acts



Need for Additional Views for Service Processes



Ontology Structure for Process Representation



Classification and Initial Support Example in OWL

```
<owl:Class rdf:ID="Task">  
  <rdfs:subClassOf rdf:resource="#Functional_View"/>  
</owl:Class>
```

```
<owl:Class rdf:ID="Activity">  
  <rdfs:subClassOf rdf:resource="#Operational_View"/>  
  <rdfs:subClassOf>  
    <owl:Class rdf:ID="View_Element"/>  
  </rdfs:subClassOf>  
</owl:Class>
```

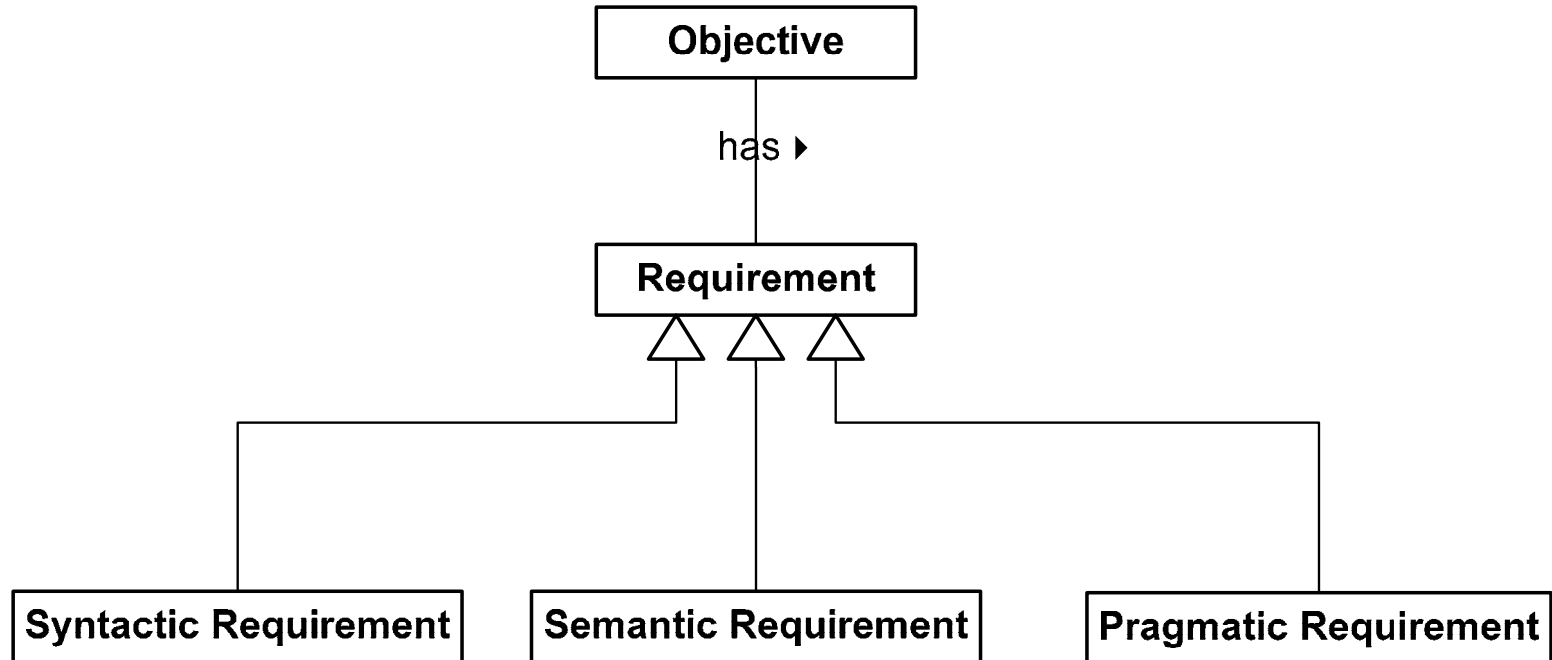
```
<owl:ObjectProperty rdf:ID="is_followed_by">  
  <rdfs:domain rdf:resource="#Activity"/>  
  <rdfs:range rdf:resource="#Activity"/>  
</owl:ObjectProperty>
```

```
<owl:FunctionalProperty rdf:ID="is_part_of">  
  <rdfs:range rdf:resource="#Task"/>  
  <rdfs:domain rdf:resource="#Activity"/>  
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>  
</owl:FunctionalProperty>
```

Classification and Initial Support Example in OWL

```
<Activity rdf:ID="Classify_incidents">
  <is_followed_by>
    <Activity rdf:ID="Match_against_known_errors_and_problems">
      <is_part_of>
        <Task rdf:ID="Classification_and_initial_support"/>
      </is_part_of>
    </is_followed_by>
    <is_followed_by>
      <Activity rdf:ID="Inform_problem_management">
        <is_part_of rdf:resource="#Classification_and_initial_support"/>
      </Activity>
    </is_followed_by>
  </Activity>
</is_followed_by>
<is_part_of rdf:resource="#Classification_and_initial_support"/>
</Activity>
```

Types of Compliance Requirements



Syntactic Requirements

- Rules *defining* which model elements have to be used or may be used for the definition of service processes.
- “Meta-model” for service process models.
- Can be easily expressed via logic operators and directly represented in an ontology
- Example: “ a responsible role is required for each activity”

$\text{activity}(a) \Rightarrow \exists r \quad \text{role}(r) \wedge \text{responsible}(a,r)$

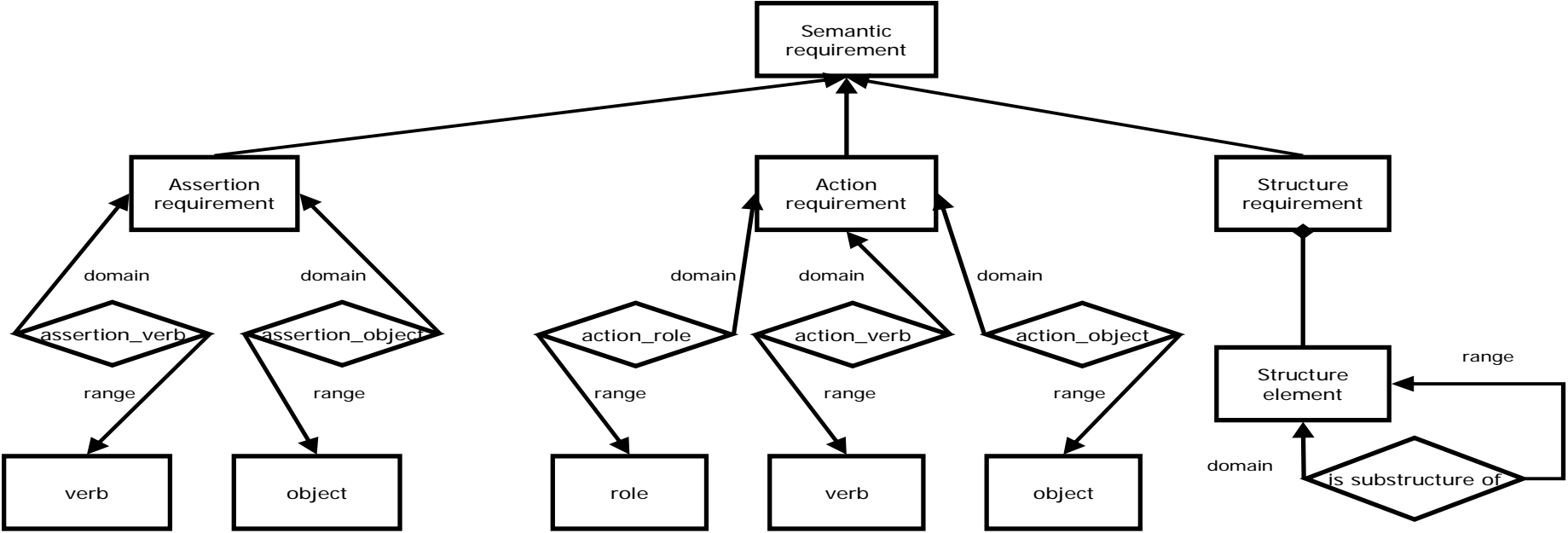
Syntactic Requirement Example

```
<owl:ObjectProperty rdf:ID="role_attachment">  
  <rdfs:range rdf:resource="#Role"/>  
  <rdfs:domain rdf:resource="#Activity"/>  
  <rdf:type  
    rdf:resource="http://www.w3.org/2002/07/owl#InverseFunc  
    tionalProperty"/>  
</owl:ObjectProperty>  
<owl:ObjectProperty rdf:ID="responsible">  
  <rdfs:subPropertyOf rdf:resource="#role_attachment"/>  
  <rdf:type  
    rdf:resource="http://www.w3.org/2002/07/owl#InverseFunc  
    tionalProperty"/>  
</owl:ObjectProperty>
```

Semantic Requirements

- *Postulate* the existence of certain structures or objects in the service process
- 3 types
 - assertion requirements: conditions to match
 - ISO 20000 example: “service levels shall be monitored”
 - action requirements: actions to be made
 - ISO 20000 example: “the management shall conduct reviews”
 - structure requirements: structures that must exist
 - E.g., ISO 20000 defines the content of a service report: it should contain the performance towards service level targets, trend information, etc.

Types of Semantic Requirements



Example Semantic Requirement: Action Requirement

```
<owl:Class rdf:ID="Requirement_Conduct_Reviews">
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:hasValue>
            <Object rdf:ID="Reviews"/>
          </owl:hasValue>
          <owl:onProperty>
            <owl:FunctionalProperty rdf:about="#action_object"/>
          </owl:onProperty>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty>
            <owl:ObjectProperty rdf:about="#action_role"/>
          </owl:onProperty>
          <owl:hasValue>
            <Role rdf:ID="Management">
              <is_member_of>
                <Section rdf:ID="Section_19"/>
              </is_member_of>
            </Role>
          </owl:hasValue>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

“the management shall
conduct reviews”

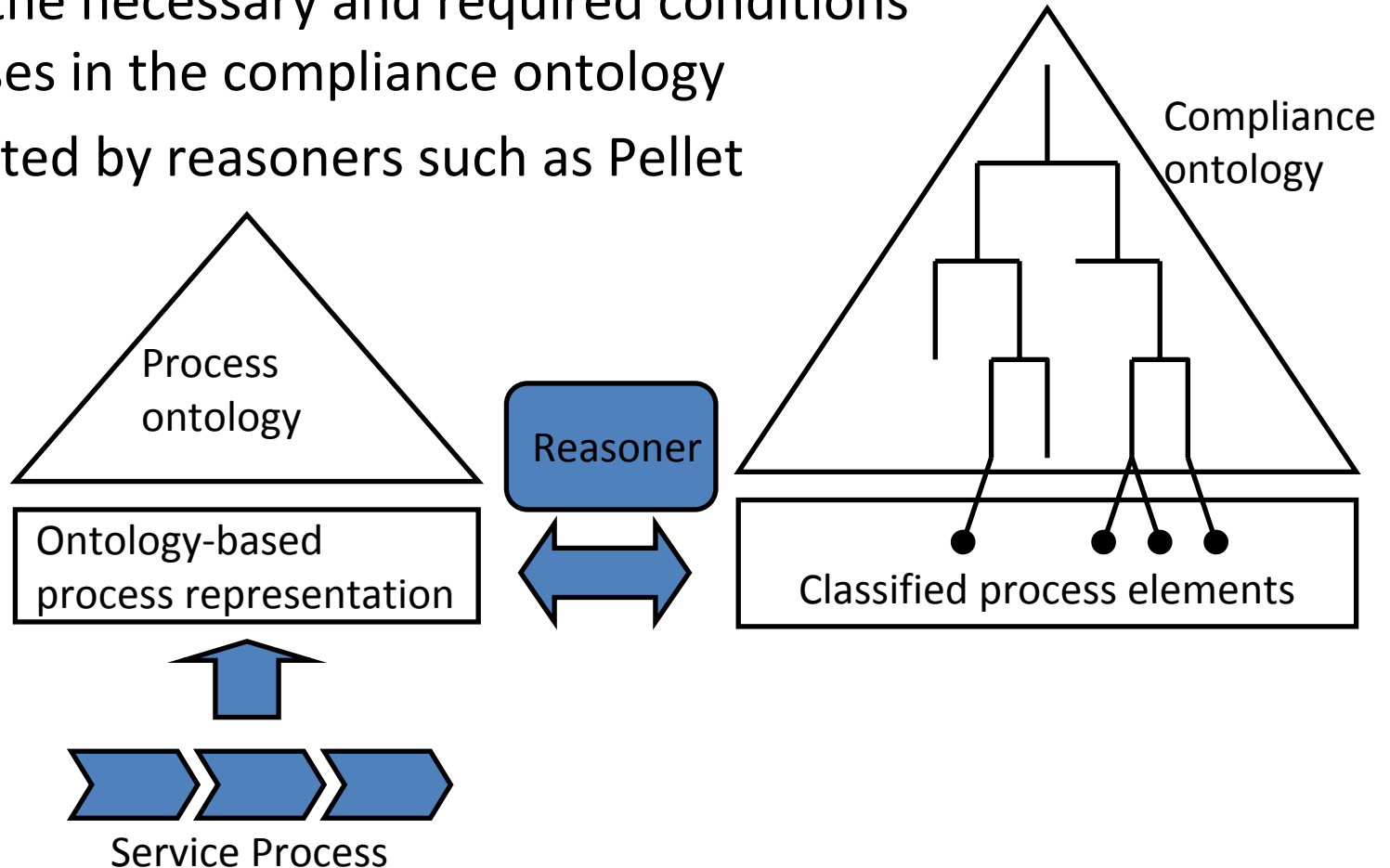
```
<owl:Restriction>
  <owl:onProperty>
    <owl:ObjectProperty rdf:about="#action_verb"/>
  </owl:onProperty>
  <owl:hasValue>
    <Verb rdf:ID="conduct"/>
  </owl:hasValue>
</owl:Restriction>
</owl:intersectionOf>
</owl:Class>
</owl:equivalentClass>
<rdfs:subClassOf rdf:resource="#Action_Requirement"/>
</owl:Class>
```

Pragmatic Requirements

- Define abstract goals to be achieved
 - ISO 20000 example: “communicate the importance of meeting service management objectives and the need for continuous improvement”
- Representation und verification difficult because the effects are outside the computer system
- Endorsement is often used to check pragmatic requirements

Checking for Compliance

- Realization is used for checking compliance: each process element is checked whether it fulfills the necessary and required conditions of classes in the compliance ontology
- Supported by reasoners such as Pellet



Summary

- The compliance of service processes to standards is essential for the quality of service processes
- Service processes require a special representation using special views
- Approach for representing compliance requirements:
 - Syntactic: define rules for service process descriptions
 - Semantic: postulate existence of objects or structures
 - Pragmatic (abstract goals): use endorsements
- Expected reduction in recurring compliance checking effort (internal and external) through ontology reuse and (semi-)automated checking by reasoners