

Taking two in tango —— Ontology and AOP

VicnentX@BJUG

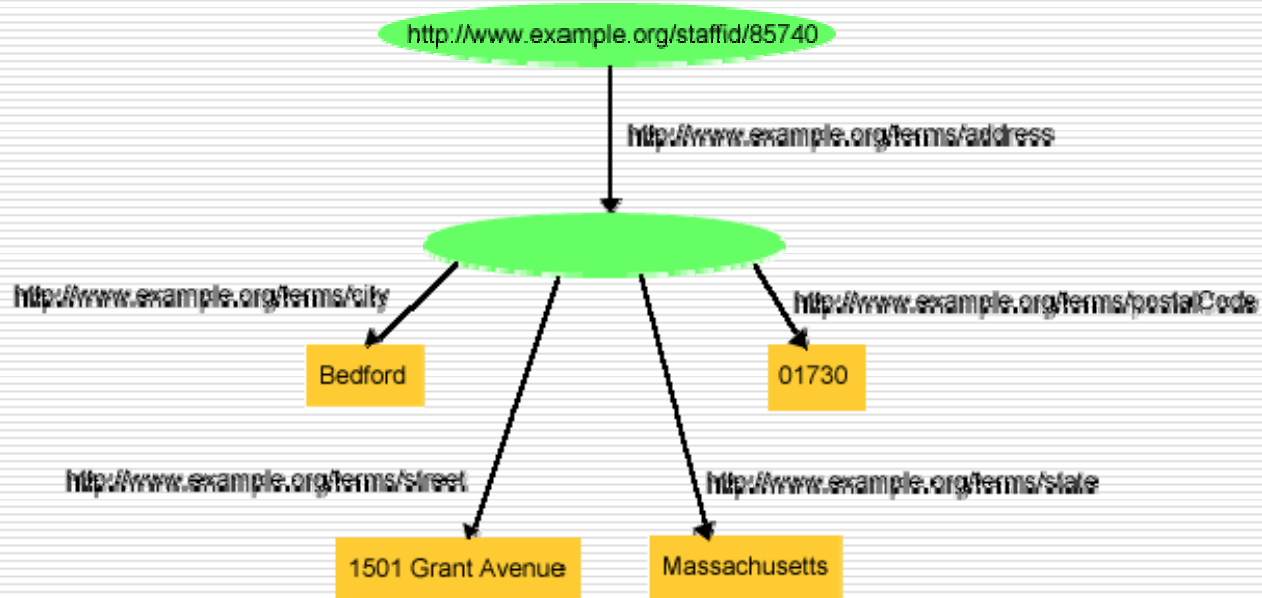
Basic of RDF Semantic

- An assertional language for expressing propositions
- Propositions is RDF triples
 - Subject
 - Predication
 - Object



Basic of RDF Semantic

□ Graph Data Model



Basic of RDF Semantic

- A graph represents a set of triple
 - <Subject, Predicate, Object>

<staffid/85740, address, _1>

<_1, city, bedford>

<_1, postalCode, 01730>

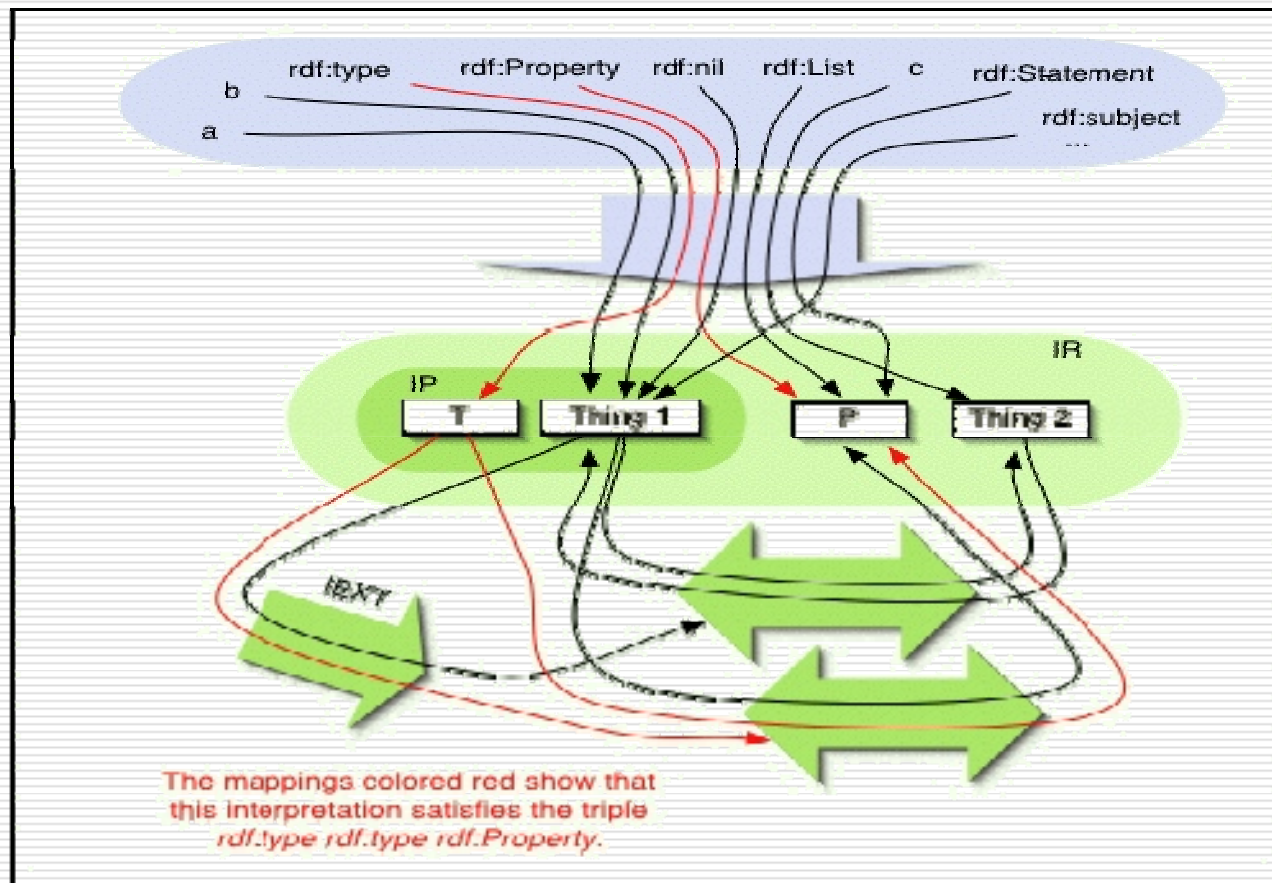
<_1, street, 1501 Grant...>

<_1, state, Massachusetts>

Basic of RDF Semantic

- asserting a sentence makes a claim about the **world**
 - the world, to be an interpretation which makes the sentence true
-

Ontology Oriented



Basic of RDF Semantic

- Anyone can make statements about any resource
 - Open World Framework
-

Property-Centric

- Domain, Range
 - RDF axiomatic triples
 - `rdfs:domain rdfs:domain rdf:Property`
 - `rdfs:range rdfs:domain rdf:Property`
 - `rdfs:subPropertyOf rdfs:domain
rdf:Property`
-

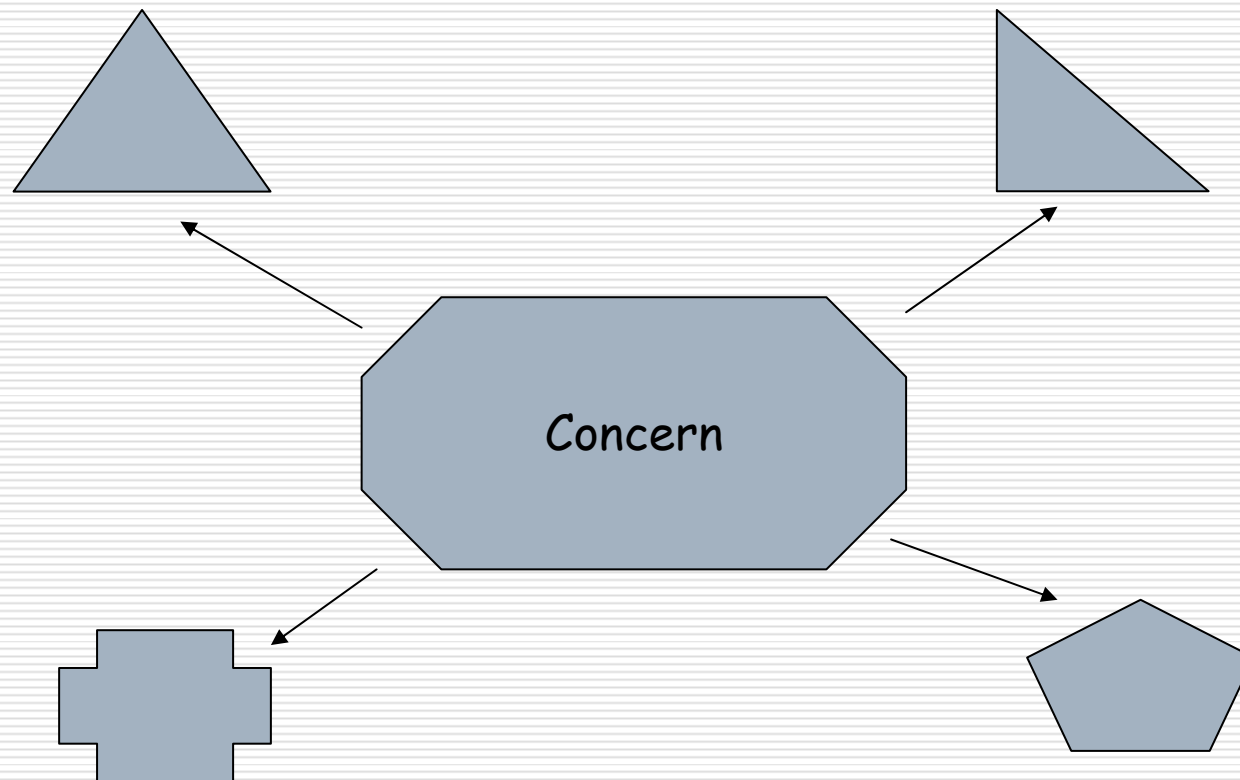
Property-Centric

- Characters of subject/object depends on their properties
 - Ontology
-

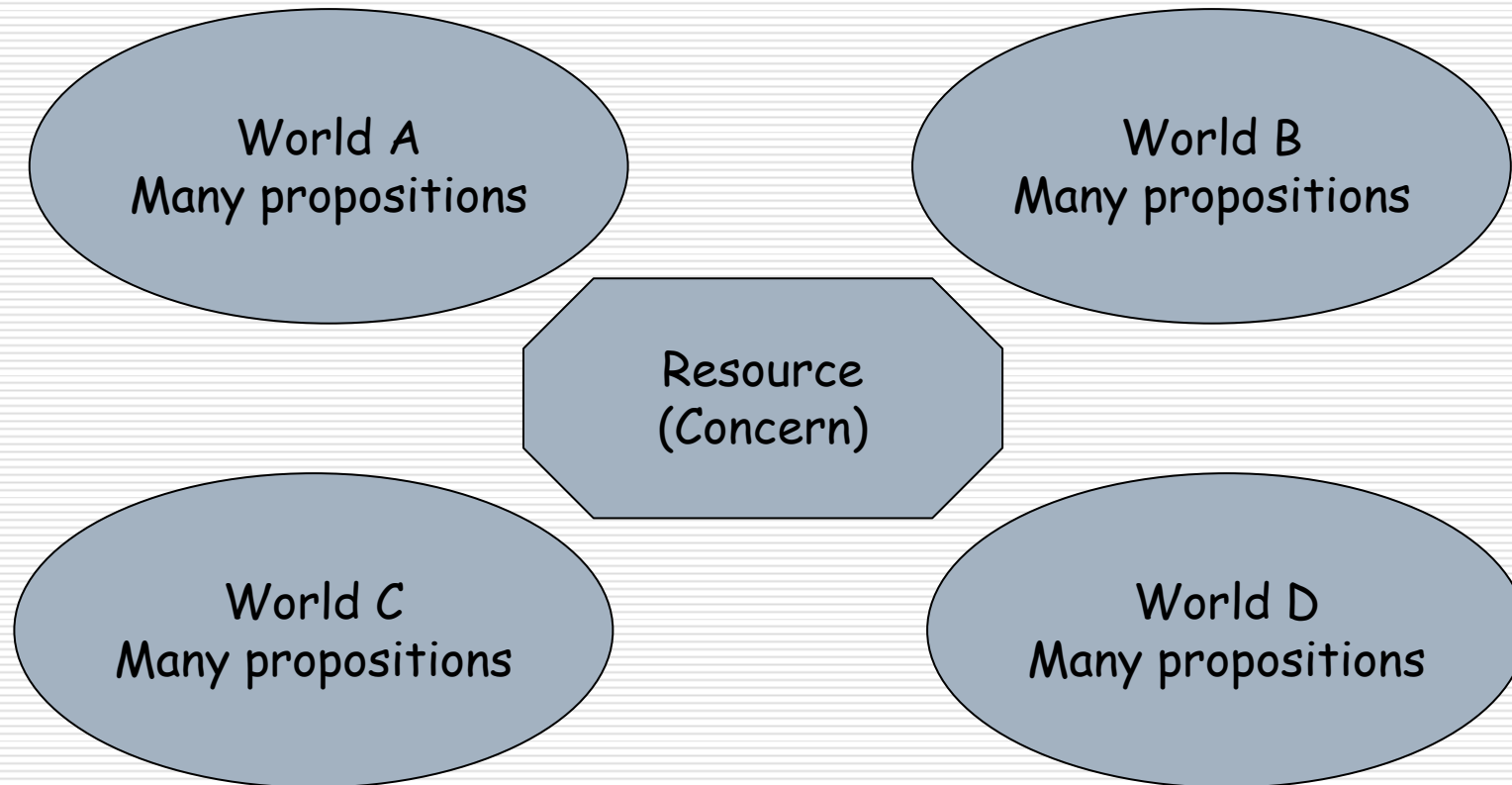
Property-Centric

- Modeling method as “property”
 - From UML/MOF perspective
 - Structural Feature
 - Behavioral Feature
-

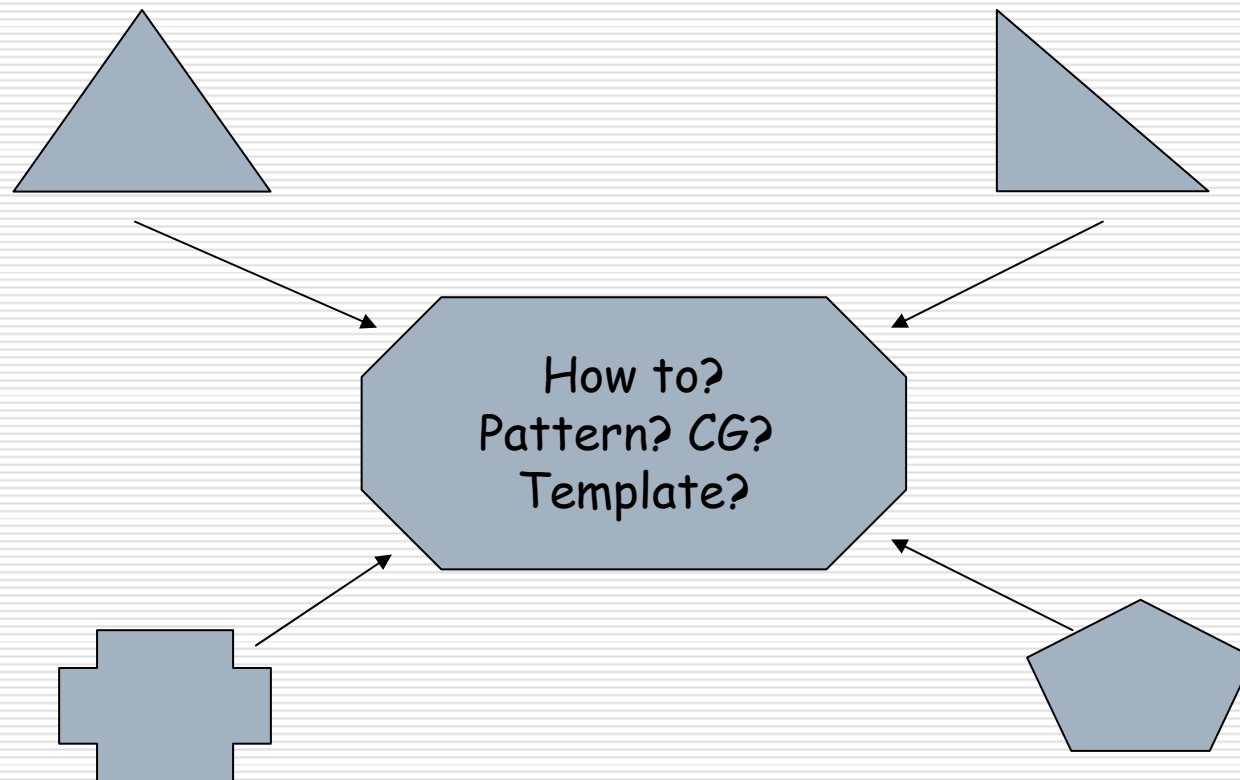
Why Ontology?



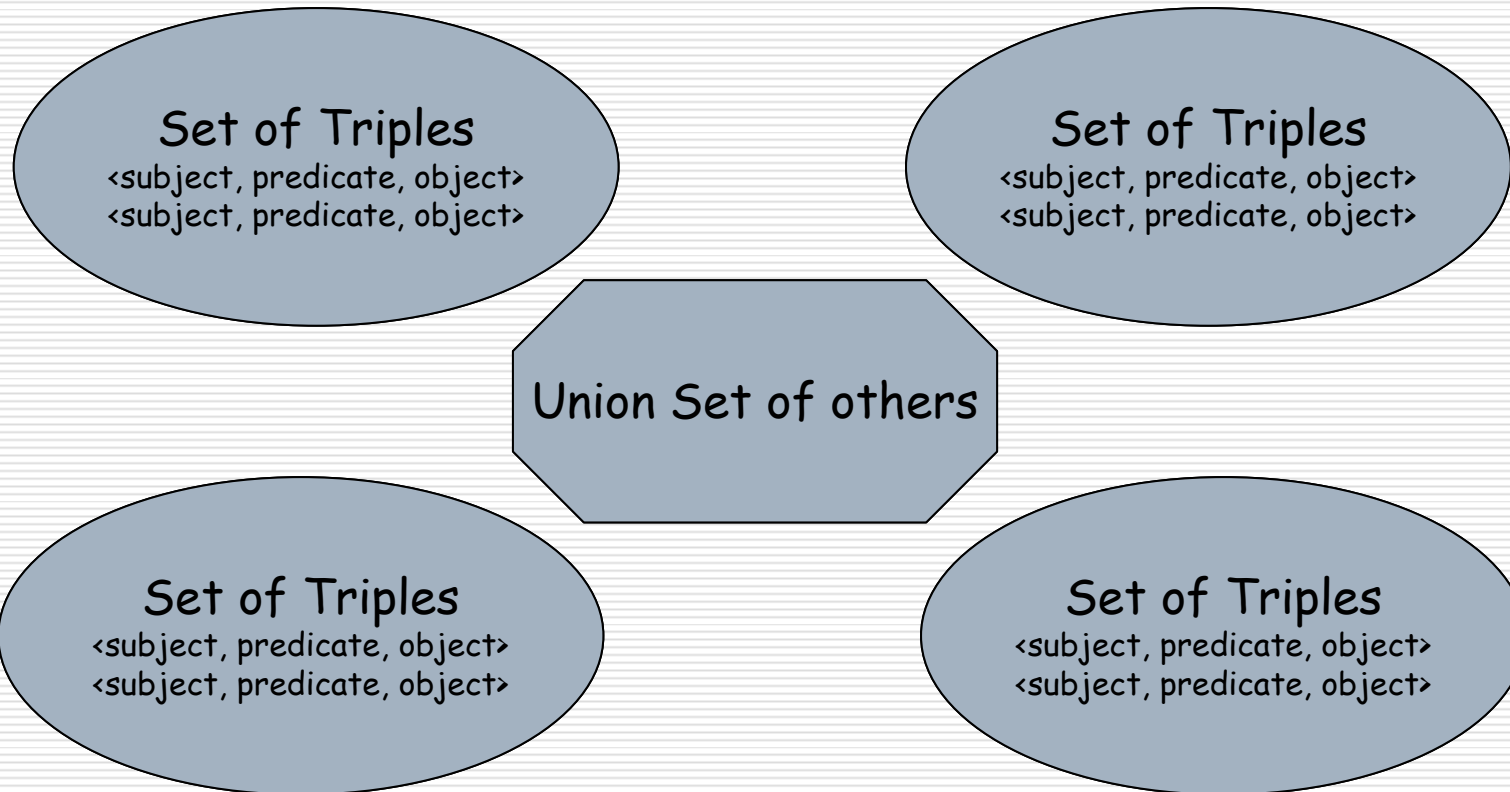
Why Ontology?



Why Ontology?



Why Ontology?



Why Ontology?

- Multi-Dimensional Separation of Concerns
 - Formal semantic
-

Age of Ontology-Oriented

- Breaking Closed-World Analysis Framework
 - Better analysis and design
 - Semantic-Oriented Composition
-

Aspect-Oriented Programming

- Revolutionary way of software composition
 - Lack of semantic-oriented composition
 - AspectJ 5 will be BIG!
-

Using Aspect as Property

- Pointcut as Domain
 - Advise as semantic interpreter
-

Using Aspect as Property

```
aspect DataProperty {  
    private String Subject.property;  
}
```

```
<Subject, Property, _1>
```

```
<_1, typeof, xsd:String>
```

```
aspect BehavioralProperty {  
    private String Subject.method() {  
        .....  
    }  
}
```

<Subject, method, ?> or
<Subject, hasMethod, method>(meaningless)

Using Aspect as Property

```
aspect TypeOfProperty {  
    declare parents : (@Subject *)  
    implements ParentType;  
}
```

```
<Subject, typeof, ParentType>
```

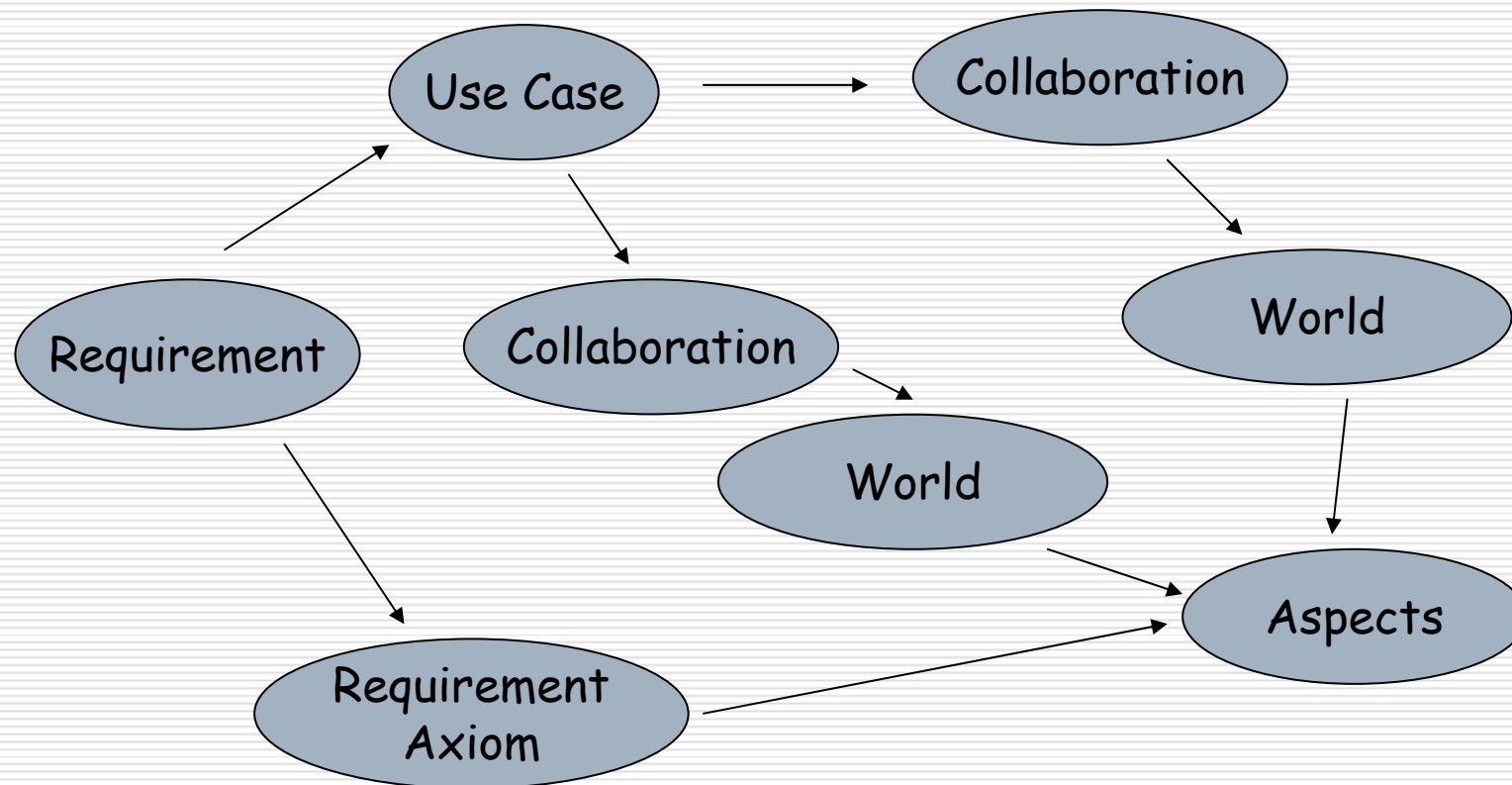
AOP for Ontology

- Lack for Range Discussion
 - Lack for semantic reason
 - Lack for Property Statement
-

AOP for Ontology

- AOP is the bridge from Object-Oriented to Ontology-Oriented
-

The Big Picture



Thanks

