

OSLC Specification for Dynamic Simulation Models

Axel Reichwein

January 14, 2014

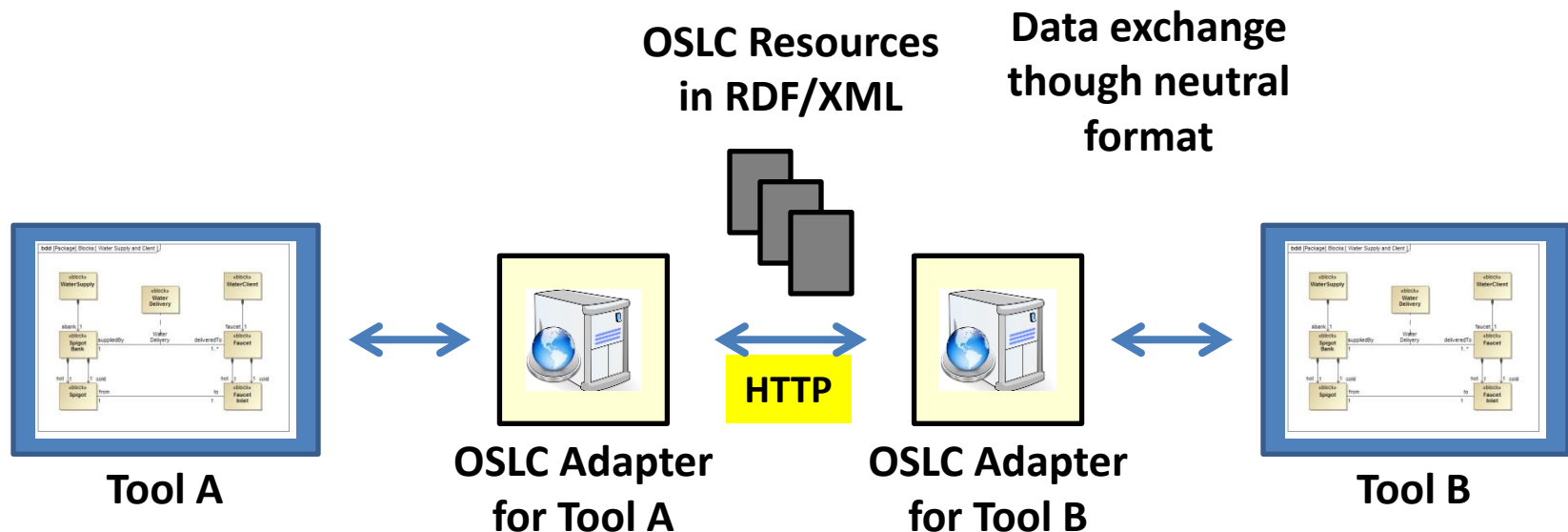
Overview

- Part I: RDF resources for defining an OSLC specification
- Part II: Common modeling concepts used in the dynamic simulation domain
- Part III: Creation of OSLC specification for dynamic simulation models

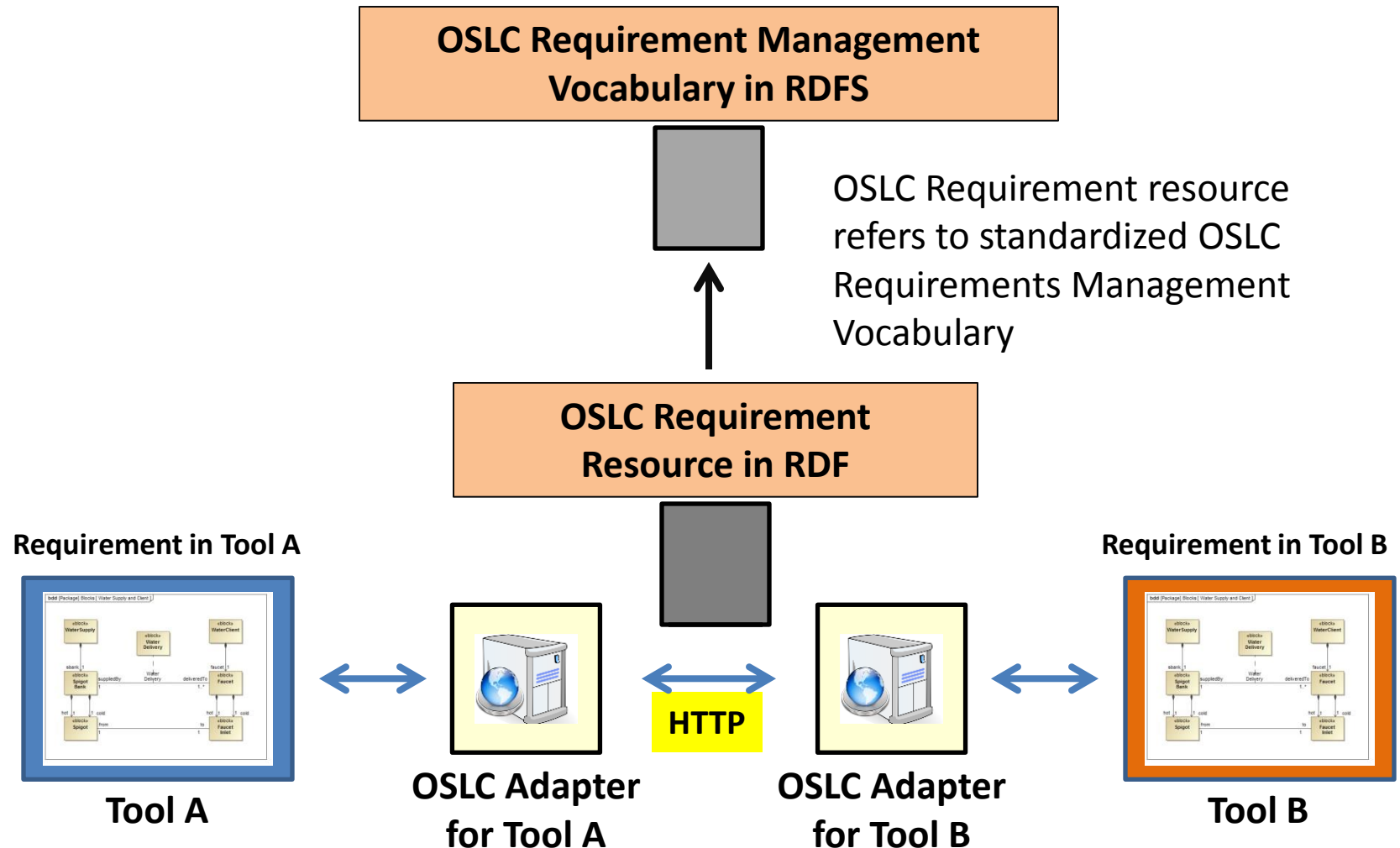
Part I: RDF Resources for Defining an OSLC Specification

Tool Interoperability through Standardized RDF Vocabularies

- OSLC specifications provide RDF vocabularies for specific domains for the purpose of supporting interoperability



Example: Standardized RDF Vocabulary for Requirements



OSLC Requirements Management Vocabulary

- <http://open-services.net/ns/rm#> redirects to <http://open-services.net/bin/view/Main/RmVocabulary#>

HTML representation of OSLC RM Vocabulary

The OSLC Requirements Management(RM) Vocabulary

The namespace URI for this vocabulary is:

```
http://open-services.net/ns/rm#
```

This page lists the RDFS classes and RDF properties that make up the OSLC vocabulary. Following [W3C best practices](#)

More details on how this page is generated and other related material can be found in the [OSLC URI Naming Guidelines](#).

Description:

This RDFS Schema defines the Open Services for Lifecycle Collaboration Requirements Management and Definition voc specifications.

See Also:

- <http://open-services.net>
- <http://open-services.net/bin/view/Main/RmHome>
- <http://open-services.net/bin/view/Main/RmSpecificationV2>

RDFS Classes in this namespace

[Requirement](#), [RequirementCollection](#)

RDF Properties in this namespace

[affectedBy](#), [elaboratedBy](#), [implementedBy](#), [specifiedBy](#), [trackedBy](#), [uses](#), [validatedBy](#)

OSLC Requirements Management Vocabulary

- <http://open-services.net/ns/rm#> + *Accept header = application/rdf+xml*

RDF/XML representation of OSLC Requirements Management Vocabulary

```
<rdfs:Class rdf:about="http://open-services.net/ns/rm#Requirement">
  <rdfs:label xml:lang="en-GB">Requirement</rdfs:label>
  <dcterms:description xml:lang="en-GB">Statement of
    need.</dcterms:description>
  <rdfs:isDefinedBy rdf:resource="http://open-services.net/ns/rm#"/>
  <dcterms:issued>2010-10-10</dcterms:issued>
  <dcterms:modified>2010-10-10</dcterms:modified>
  <oslc:hasBasicShape
    rdf:resource="http://open-services-net/shapes/rm#requirementShape"/>
```

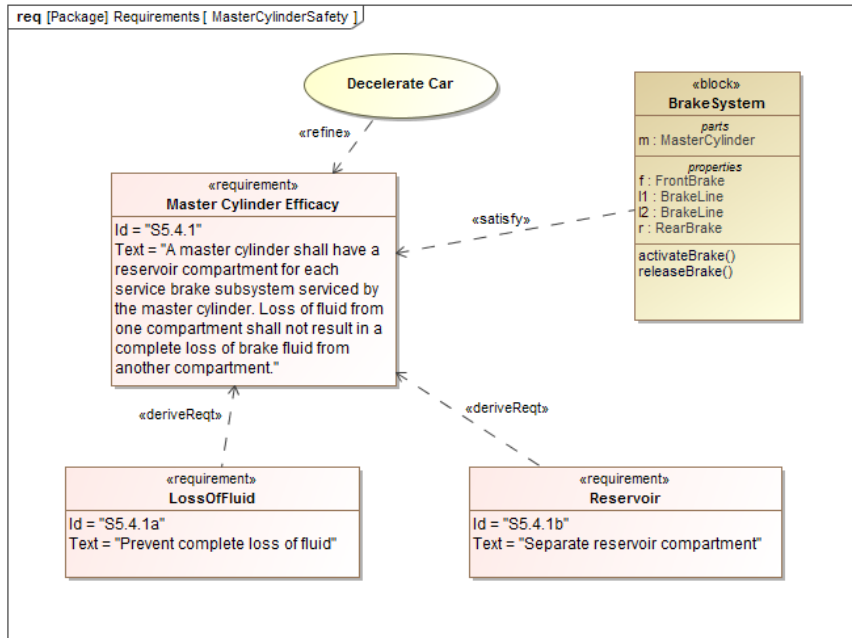
...just a snippet...

```
<rdf:Property rdf:about="http://open-services.net/ns/rm#elaboratedBy">
  <rdfs:label>elaboratedBy</rdfs:label>
  <rdfs:comment xml:lang="en-GB">An entity which elaborates.</rdfs:comment>
  <dcterms:description xml:lang="en-GB">Expresses an elaboration relationship
    between entities. For example, a model element can elaborate a
    requirement.</dcterms:description>
  <rdfs:isDefinedBy rdf:resource="http://open-services.net/ns/rm#"/>
  <dcterms:issued>2008-01-14</dcterms:issued>
  <dcterms:modified>2008-01-14</dcterms:modified>
  <rdfs:seeAlso
    rdf:resource="http://open-services.net/bin/view/Main/RmSpecificationV2#RequirementResource"/>
</rdf:Property>
```

...just a snippet...

Example of Standardized RDF Properties

Relationships in SysML



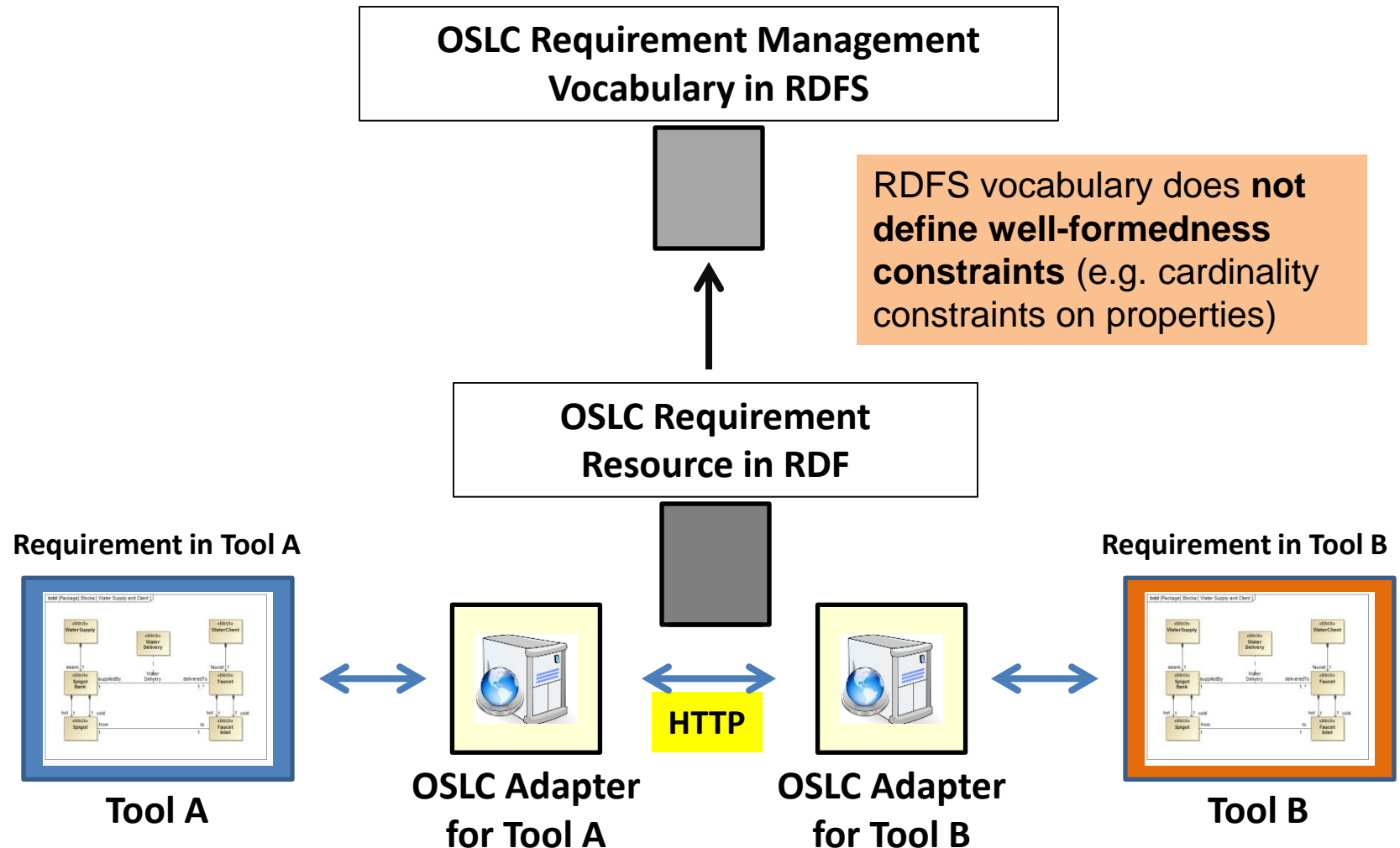
Standardized resource properties enable interoperability

Relationships in RDF

Subject: Requirement
„Master Cylinder Efficacy“

Predicate	Object
elaboratedBy	Use Case „Decelerate Car“
satisfiedBy	Block „Brake System“
derivedRqt	Requirement „Loss of Fluid“
derivedRqt	Requirement „Reservoir“

Standardized RDF Vocabularies are not enough for Tool Interoperability!

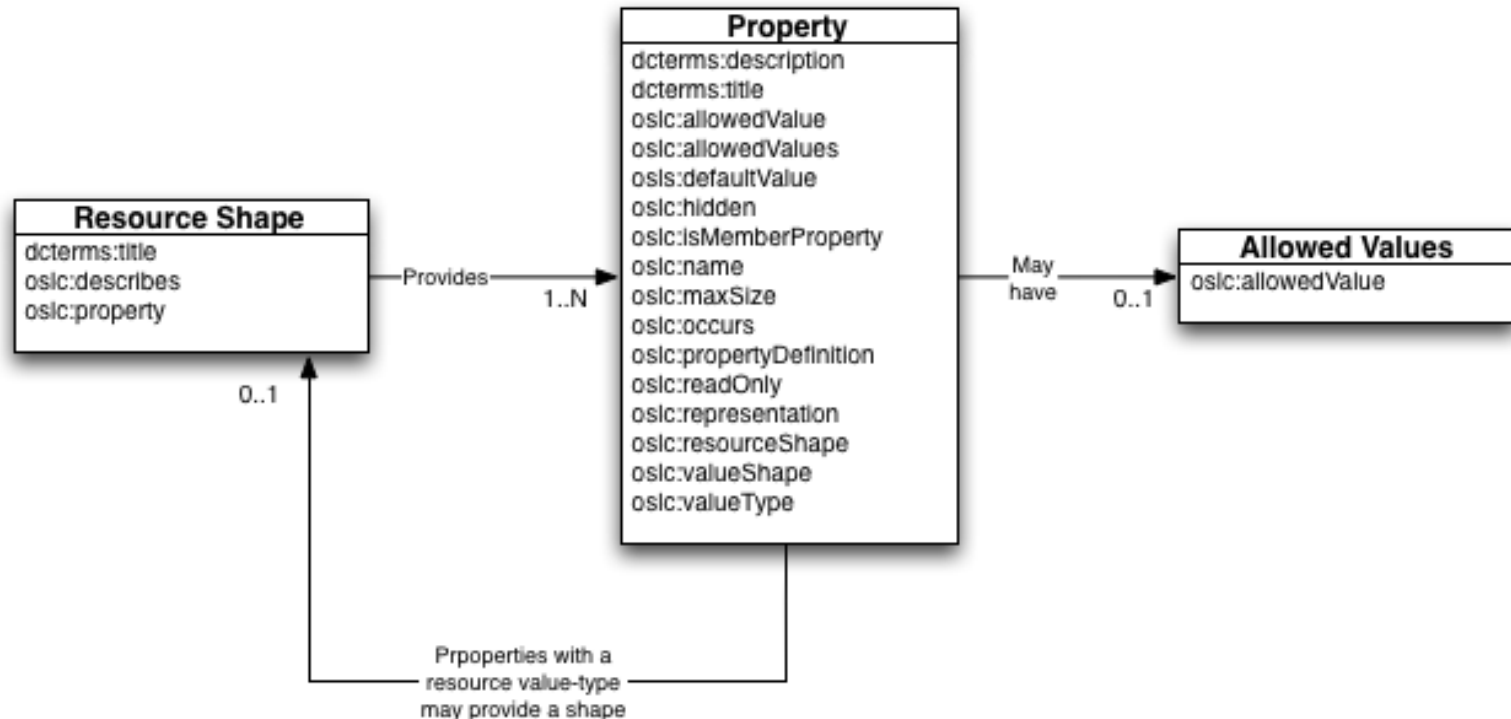


Additional RDF resources for defining constraints on RDF resources: OSLC Resource Shapes

- **RDFS Vocabulary** cannot define constraints on RDF data
- **OSLC Core vocabulary** includes additional RDFS classes and RDF properties for defining constraints on RDF data such as:
 - RDFS class *oslc:ResourceShape*
 - RDFS class *oslc:AllowedValues*
 - RDF property *oslc:occurs*
 - RDF property *oslc:allowedValue*
- **OSLC resource shapes** are RDF resources that define constraints on RDF data by using the OSLC Core vocabulary

OSLC Resource Shape

A resource shape describes constraints on properties of resources of a specific type (e.g. property value-type, property cardinality, property allowed values)



See OSLC 2.0 Appendix A Common Properties for the complete specification: http://open-services.net/bin/view/Main/OSLCCoreSpecAppendixA?sortcol=table;up=#osc_ResourceShape_Resource

OSLC Resource Shapes

- **OSLC Resource Shape = RDF vocabulary that can be used for specifying and validating constraints on OSLC resources**
- Various Resource Shapes have been developed for various domains (Change Request, Test Case, Requirement, Performance Monitoring Record)
- Arthur G. Ryman, Arnaud J Le Hors, Steve Speicher, ***OSLC Resource Shape, A language for defining constraints on Linked Data***, Rio de Janeiro, Brazil, LDOW2013
<http://events.linkedata.org/ldow2013/papers/ldow2013-paper-02.pdf>

OSLC Resource Shape for Requirements

RDF resource of type ResourceShape (URI = <http://open-services.net/ns/core#ResourceShape>)



```
<oslc:ResourceShape rdf:about="https://myDomain.com/myTool/shapes/ShapeID23">
  <oslc:describes rdf:resource="http://open-services.net/ns/rm#Requirement"/>
  <oslc:property>
    <oslc:Property>
      <oslc:name>satisfiedBy</oslc:name>
      <oslc:propertyDefinition rdf:resource="http://open-service.net/ns/rm#satisfiedBy"/>
      <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      <oslc:range rdf:resource="http://open-services.net/ns/core#Any"/>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
```

...just a snippet...

OSLC Resource Shape for Requirements

URI of resource shape (resource shape typically hosted by tool-specific OSLC service provider)



```
<oslc:ResourceShape rdf:about="https://myDomain.com/myTool/shapes/ShapeID23">
  <oslc:describes rdf:resource="http://open-services.net/ns/rm#Requirement"/>
  <oslc:property>
    <oslc:Property>
      <oslc:name>satisfiedBy</oslc:name>
      <oslc:propertyDefinition rdf:resource="http://open-service.net/ns/rm#satisfiedBy"/>
      <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      <oslc:range rdf:resource="http://open-services.net/ns/core#Any"/>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
```

...just a snippet...

OSLC Resource Shape for Requirements

oslc:describes is a property indicating that the resource shape applies to resources which are instances of this resource type (e.g. *http://open-services.net/ns/rm#Requirement*)



```
<oslc:ResourceShape rdf:about="https://myDomain.com/myTool/shapes/ShapeID23">
  <oslc:describes rdf:resource="http://open-services.net/ns/rm#Requirement"/>
  <oslc:property>
    <oslc:Property>
      <oslc:name>satisfiedBy</oslc:name>
      <oslc:propertyDefinition rdf:resource="http://open-service.net/ns/rm#satisfiedBy"/>
      <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      <oslc:range rdf:resource="http://open-services.net/ns/core#Any"/>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
```

...just a snippet...

OSLC Resource Shape for Requirements

oslc:property is an RDF property to describe a resource shape property and associated constraints

```
<oslc:ResourceShape rdf:about="https://myDomain.com/myTool/shapes/ShapeID23">
  <oslc:describes rdf:resource="http://open-services.net/ns/rm#Requirement"/>
  <oslc:property>
    <oslc:Property>
      <oslc:name>satisfiedBy</oslc:name>
      <oslc:propertyDefinition rdf:resource="http://open-service.net/ns/rm#satisfiedBy"/>
      <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      <oslc:range rdf:resource="http://open-services.net/ns/core#Any"/>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
```

...just a snippet...

Nested oslc:property in RDF/XML due to "striping" as described at <http://www.w3.org/2001/10/stripes/>

OSLC Resource Shape for Requirements

```
<oslc:ResourceShape rdf:about="https://myDomain.com/myTool/shapes/ShapeID23">
  <oslc:describes rdf:resource="http://open-services.net/ns/rm#Requirement"/>
  <oslc:property>
    <oslc:Property>
      <oslc:name>satisfiedBy</oslc:name>
      <oslc:propertyDefinition rdf:resource="http://open-service.net/ns/rm#satisfiedBy"/>
      <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      <oslc:range rdf:resource="http://open-services.net/ns/core#Any"/>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
```

...just a snippet...

oslc:propertyDefinition indicates the URI of the property whose usage is being constrained

OSLC Resource Shape for Requirements

```
<oslc:ResourceShape rdf:about="https://myDomain.com/myTool/shapes/ShapeID23">
  <oslc:describes rdf:resource="http://open-services.net/ns/rm#Requirement"/>
  <oslc:property>
    <oslc:Property>
      <oslc:name>satisfiedBy</oslc:name>
      <oslc:propertyDefinition rdf:resource="http://open-service.net/ns/rm#satisfiedBy"/>
      <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      <oslc:range rdf:resource="http://open-services.net/ns/core#Any"/>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
```

...just a snippet...

oslc:occurs indicates the cardinality of the constrained RDF property

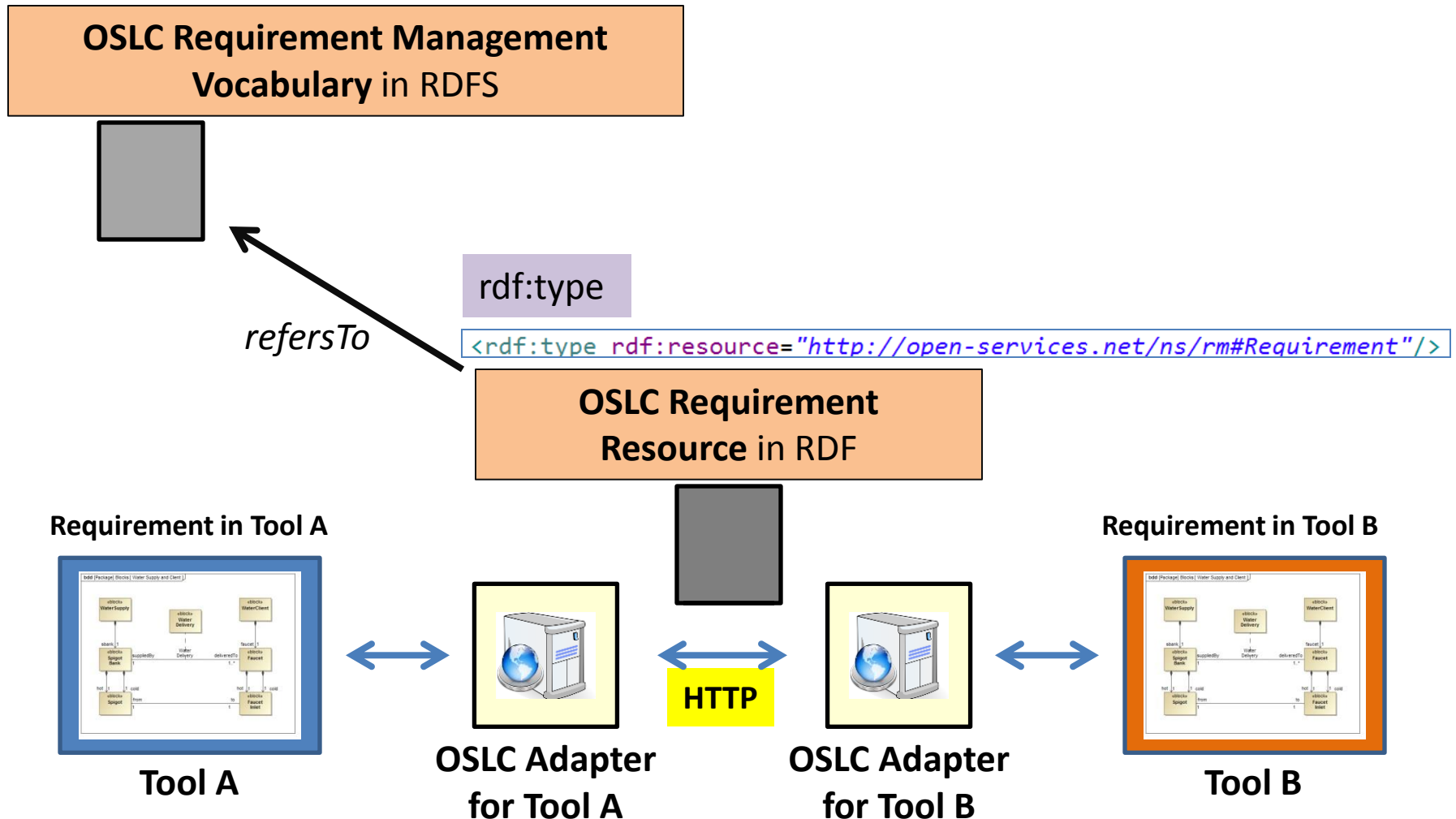
OSLC Resource Shape for Requirements

```
<oslc:ResourceShape rdf:about="https://myDomain.com/myTool/shapes/ShapeID23">
  <oslc:describes rdf:resource="http://open-services.net/ns/rm#Requirement"/>
  <oslc:property>
    <oslc:Property>
      <oslc:name>satisfiedBy</oslc:name>
      <oslc:propertyDefinition rdf:resource="http://open-service.net/ns/rm#satisfiedBy"/>
      <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      <oslc:range rdf:resource="http://open-services.net/ns/core#Any"/>
    </oslc:Property>
  </oslc:property>
</oslc:ResourceShape>
```

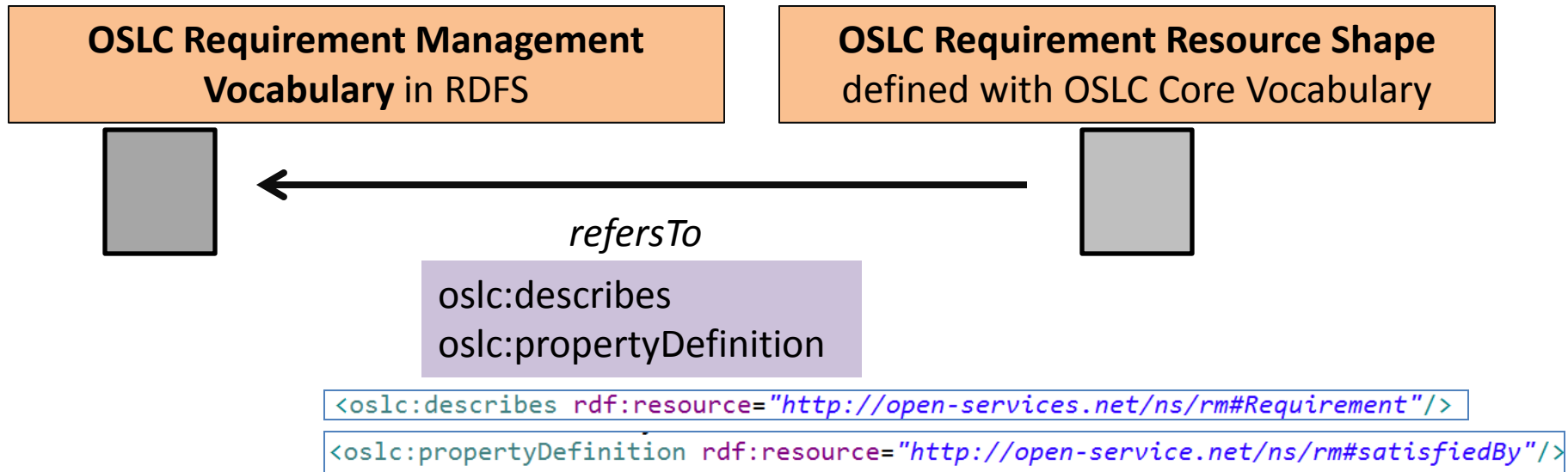
...just a snippet...

oslc:range specifies the range of possible resource types allowed

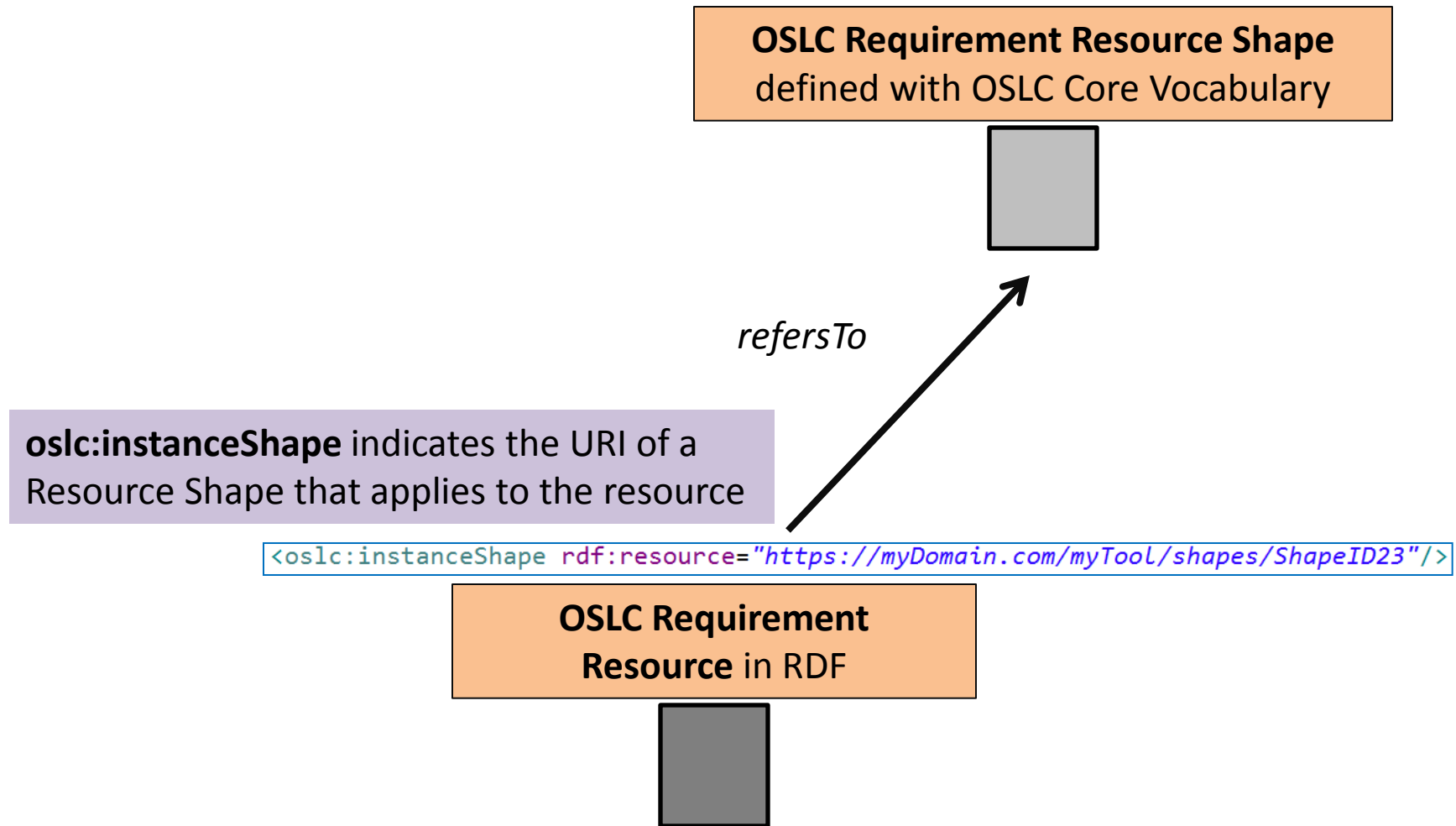
Links between RDF Resources in OSLC Interoperability Scenario



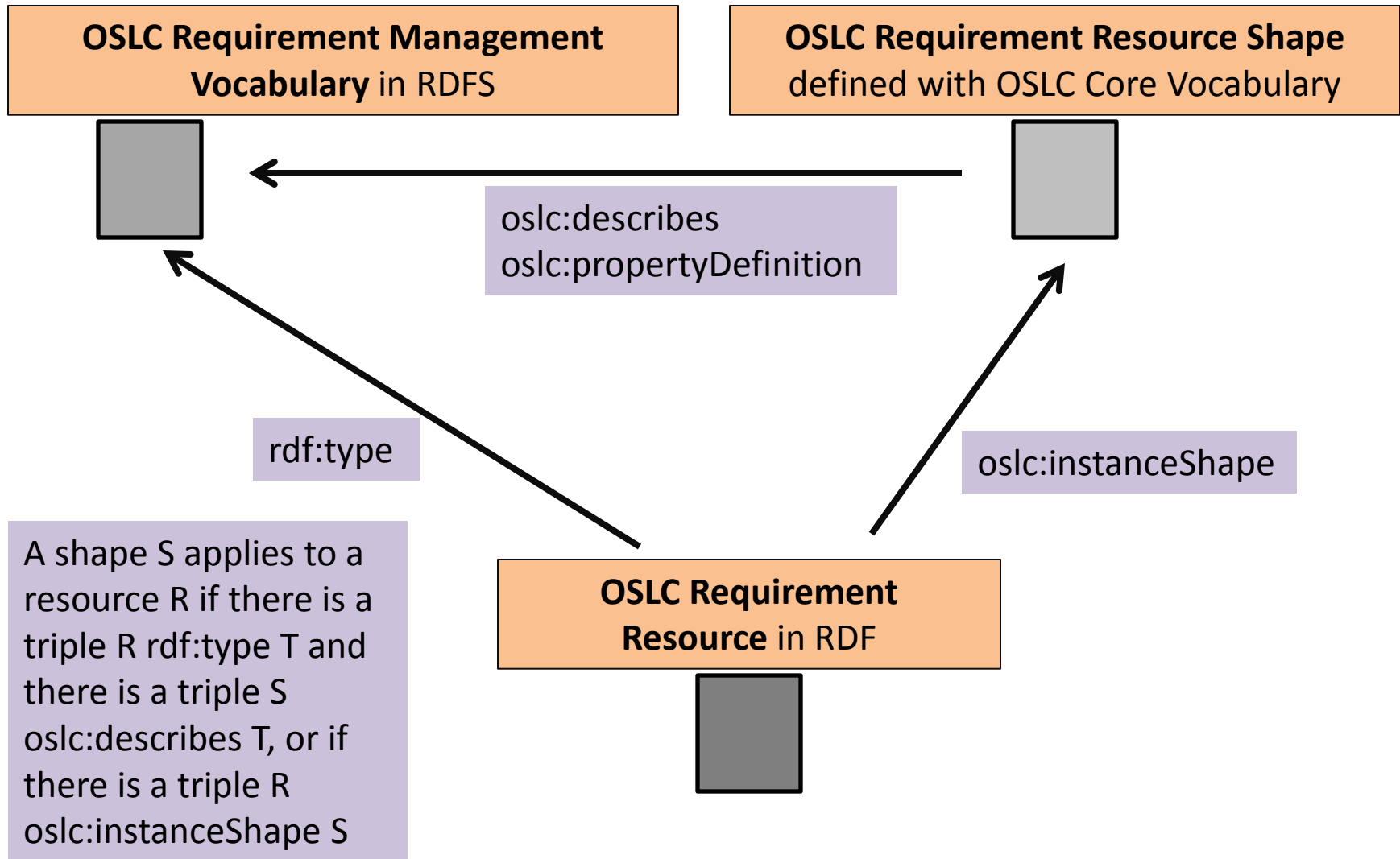
Links between RDF Resources in OSLC Interoperability Scenario



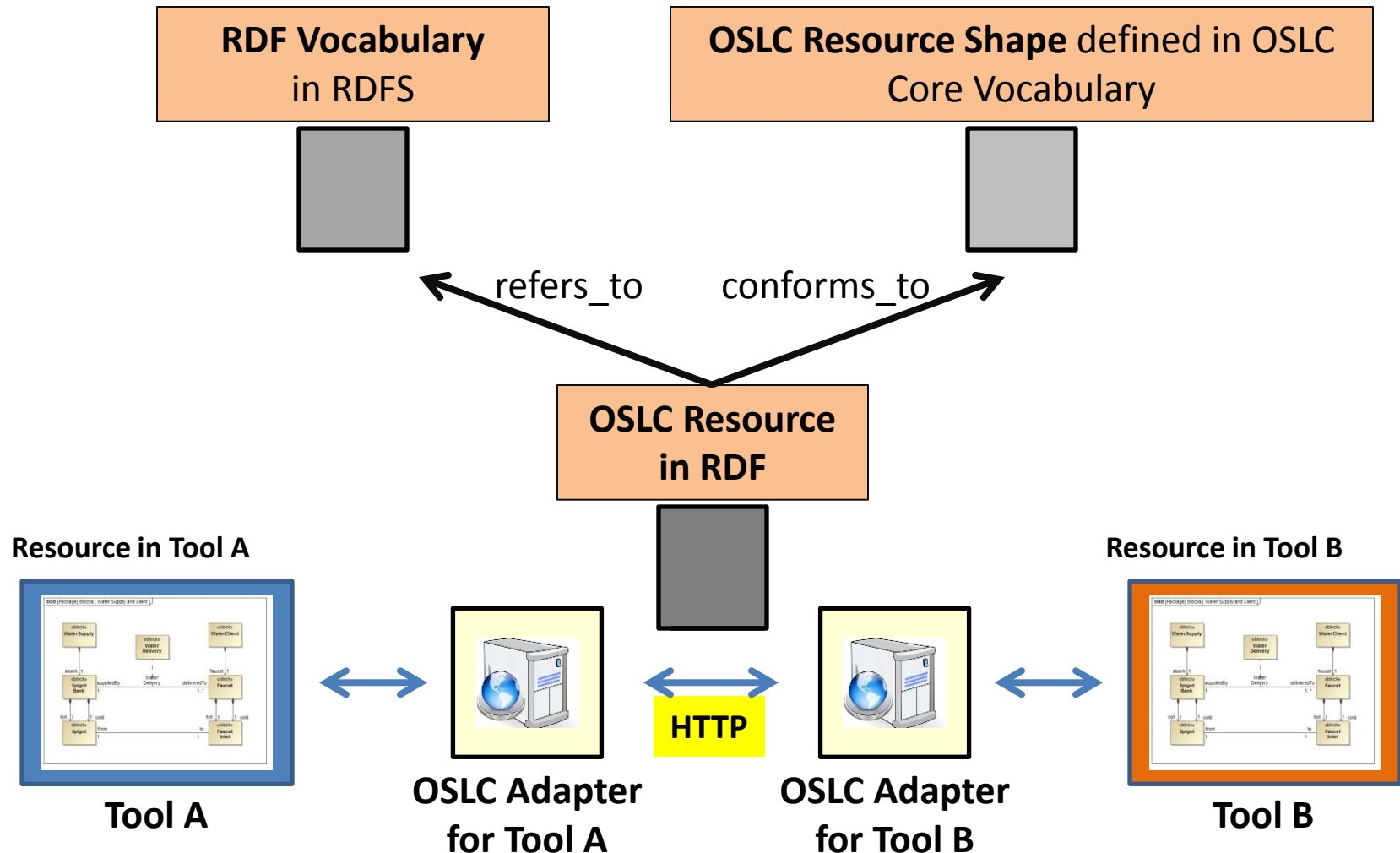
Links between RDF Resources in OSLC Interoperability Scenario



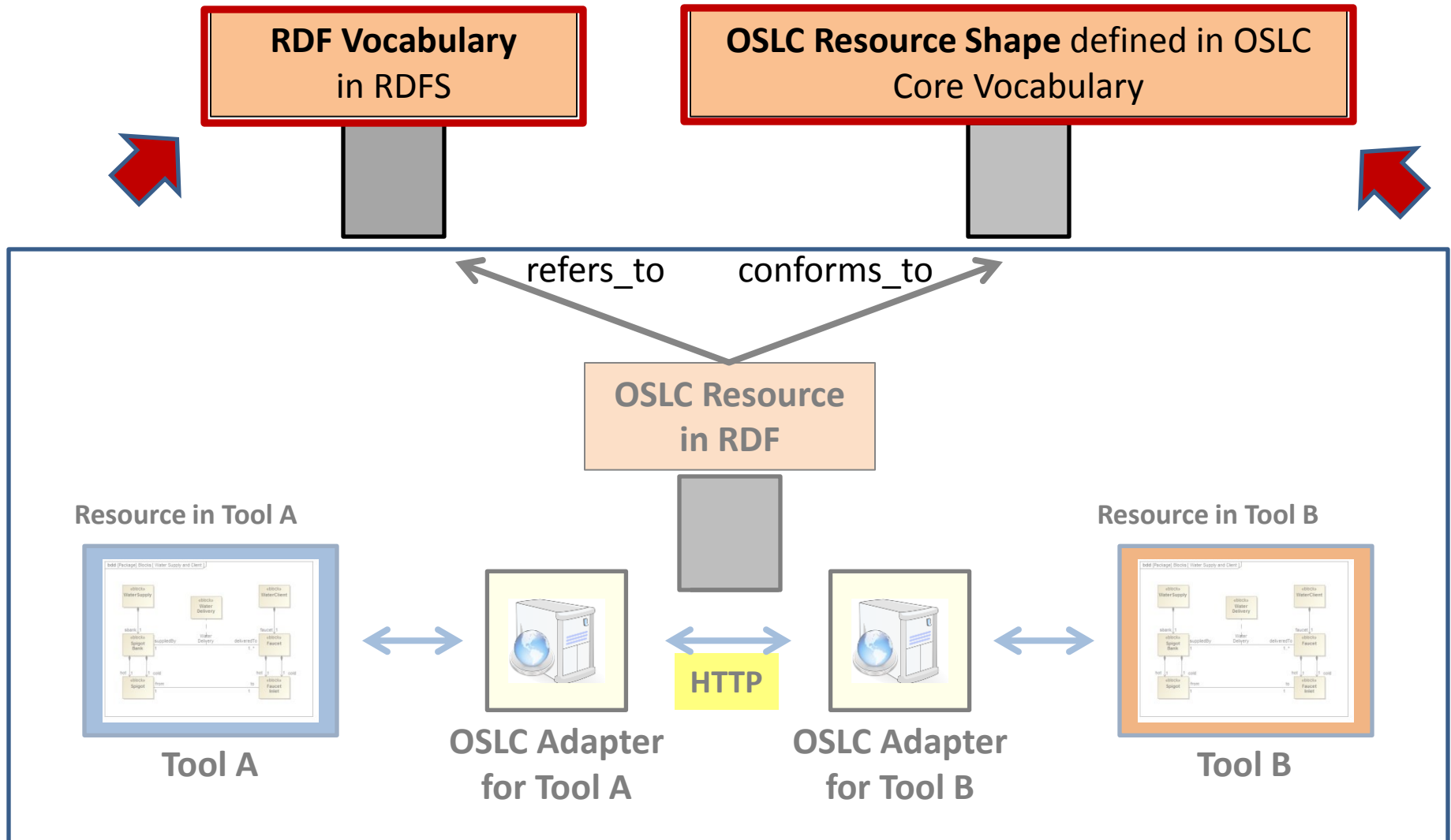
Overview of Links between RDF Resources in OSLC Interoperability Scenario



General Overview of RDF Resources for OSLC Data Interchange



Required RDF Resources for Defining an OSLC Specification



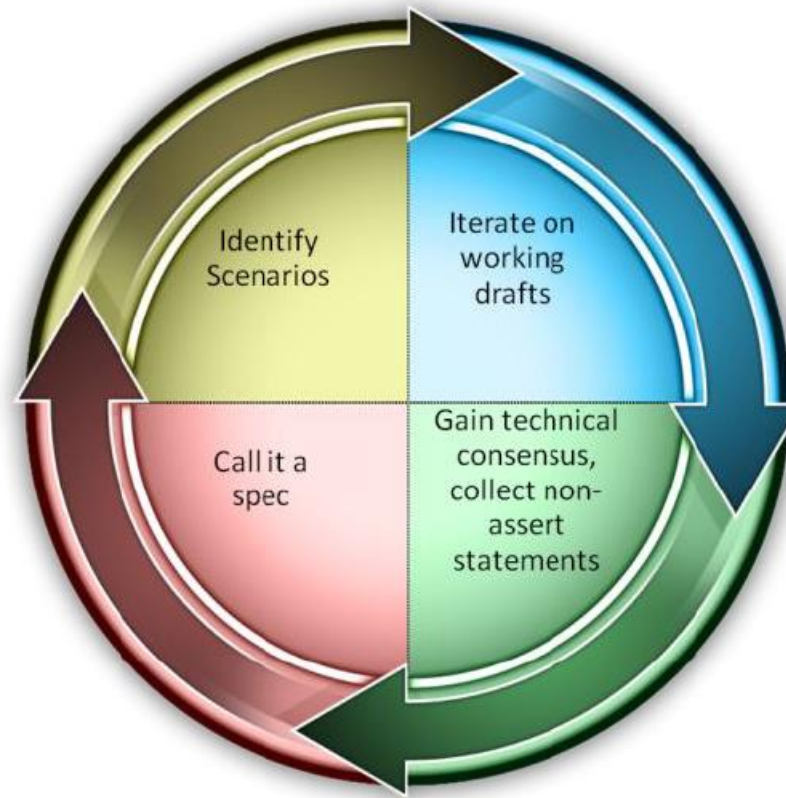
Example OSLC Requirements Management Specification Resources

OSLC Requirements Management Specification Resources	Links to Resources
Namespace URI of OSLC Requirements Management vocabulary	http://open-services.net/ns/rm
OSLC Requirements Management vocabulary in RDF/XML	http://open-services.net/ns/rm
OSLC Requirements Management vocabulary in HTML	http://open-services.net/bin/view/Main/RmSpecificationV2
OSLC Requirements Management resource shapes in RDF/XML	http://open-services.net/bin/view/Main/RmSpecificationV2Shapes

Notes on OSLC Specifications

- Minimalistic in order to support the most common scenarios
- Extensible in order to support additional scenarios
- Stable (New specification versions which include only additive changes are ok for clients. New versions including not only additive changes require the definition of a new version-specific namespace URI for the new OSLC specification)
- Reuse existing RDF vocabularies like Dublin Core
- Define resources as granular as possible in order to easily link data from different contexts
- Additional guidelines on creating OSLC specifications can be found at <http://open-services.net/bin/view/Main/ResourceGuidelines>

Development of OSLC Specifications



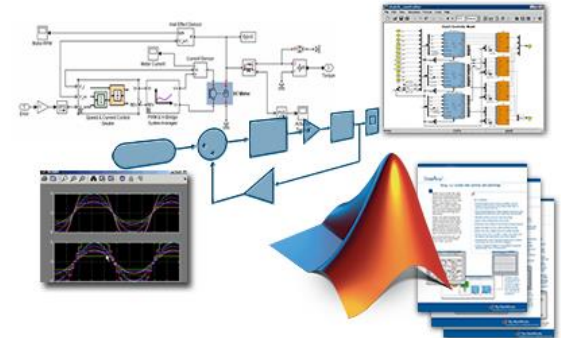
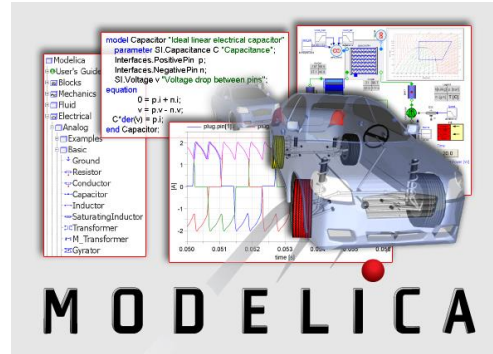
Part II: Common Modeling Concepts used in the Dynamic Simulation Domain

Scope of Investigated **Dynamic Simulation** Domain

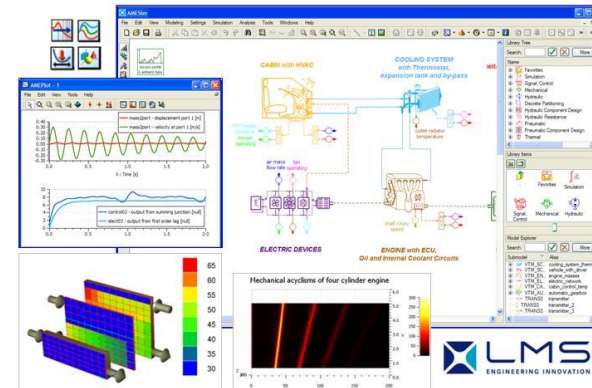
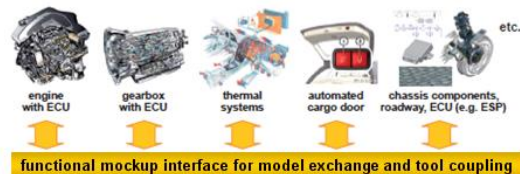
- Dynamic simulation models consisting of
 - Continuous time models
 - Differential-algebraic equations expressed in the form $f(\dot{x}, x, t) = 0$
 - Block-based modeling with signal flow and physical connections
 - Typically used for simulating control software, multi-body systems, and electric systems
- Not within scope
 - Discrete systems (e.g. state machine diagrams)
 - Partial-differential equations

Typical Dynamic Simulation Tools in Investigated Domain

- Modelica
- Simulink & SimScape
- AMESim
- FMI
- And more...



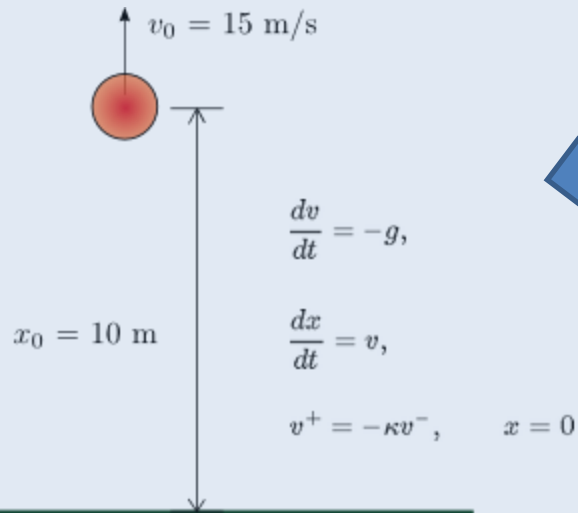
Functional Mock-up Interface



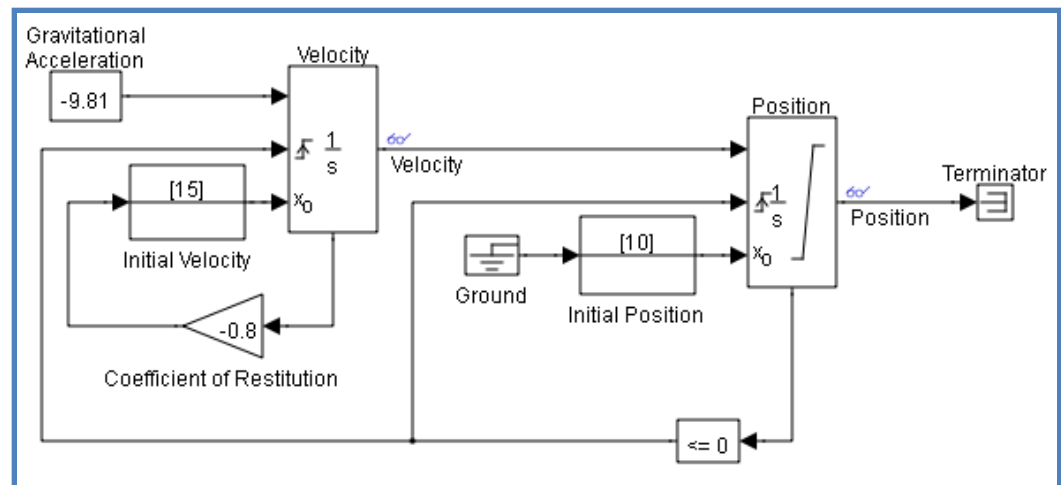
Modeling Methodologies in Investigated **Dynamic Simulation** Domain

- Signal-flow modeling
 - CAUSAL modeling style
 - Connections between blocks are directed
 - Connections between blocks describe equality constraints
 - Typically used for control engineering and signal processing
- Physical-interaction modeling
 - ACAUSAL modeling style
 - Connections between blocks are NOT directed
 - Connections between blocks describe equality AND sum-to-zero equations based on the principle of power continuity
 - Typically used for the simulation of physical systems, such as electrical or multi-body systems

Signal-Flow Modeling Example: Bouncing Ball



Block Diagram in Simulink

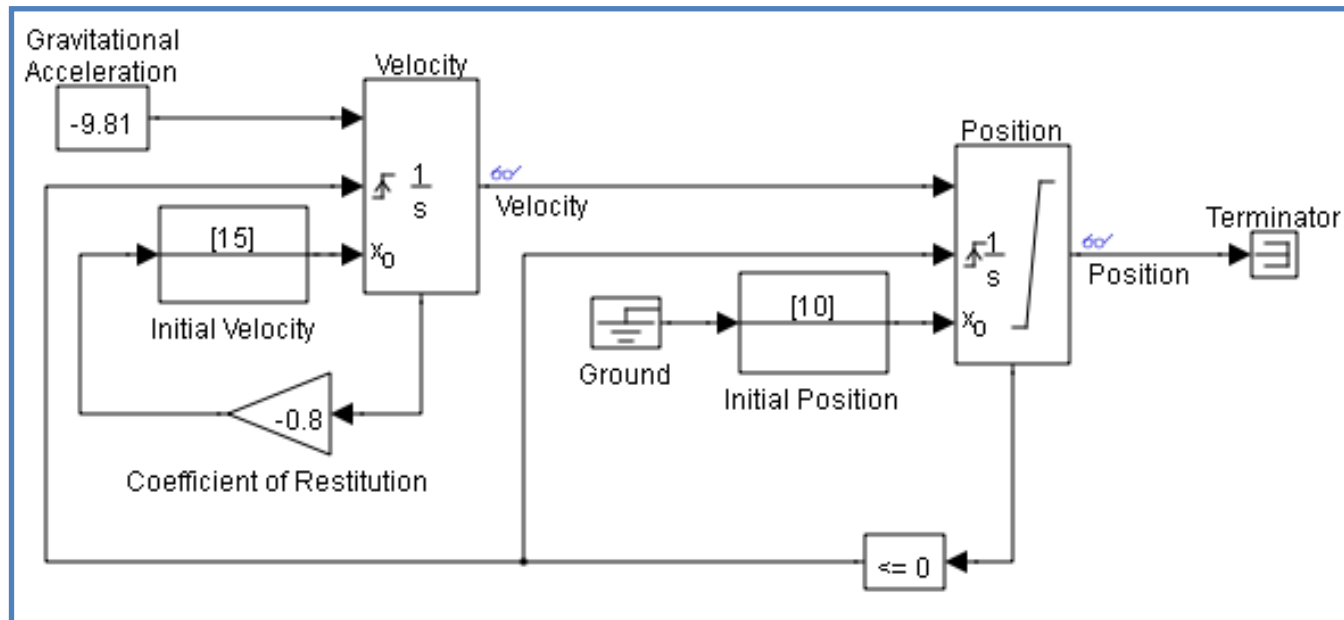


Reference: http://www.mathworks.com/products/simulink/examples.html?file=/products/demos/shipping/simulink/sldemo_bounce.html

Signal-Flow Modeling Example: Bouncing Ball

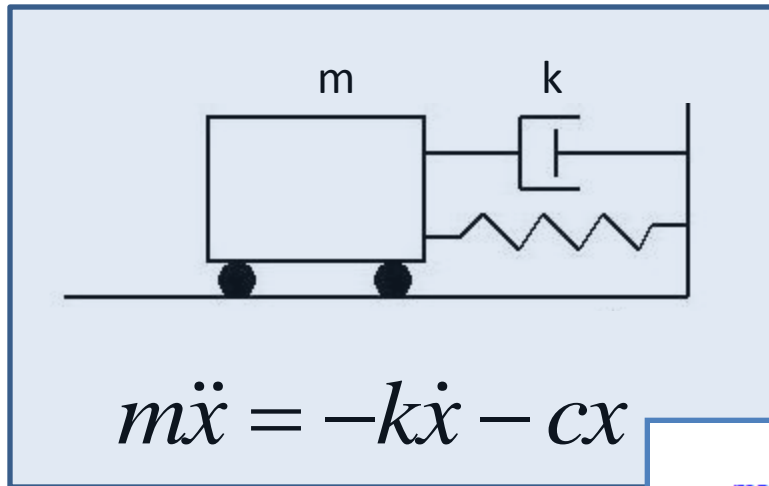
Blocks have inputs and outputs. Blocks compute outputs based on inputs. Computation performed by blocks depends on their type and their parameter values.

Block inputs and outputs go through ports. Ports are not visible in this diagram.

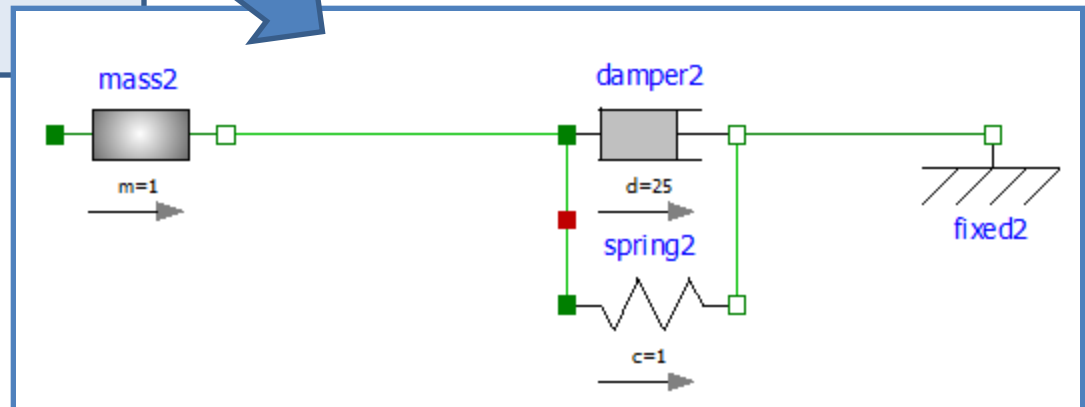


Connections represent **CAUSAL signal flows** between blocks. Signals connect block outputs with block inputs. Signals indicate **equality** between connected block outputs and inputs.

Physical-Interaction Modeling Example: Mass-Spring-Damper System



Connection Diagram in Modelica

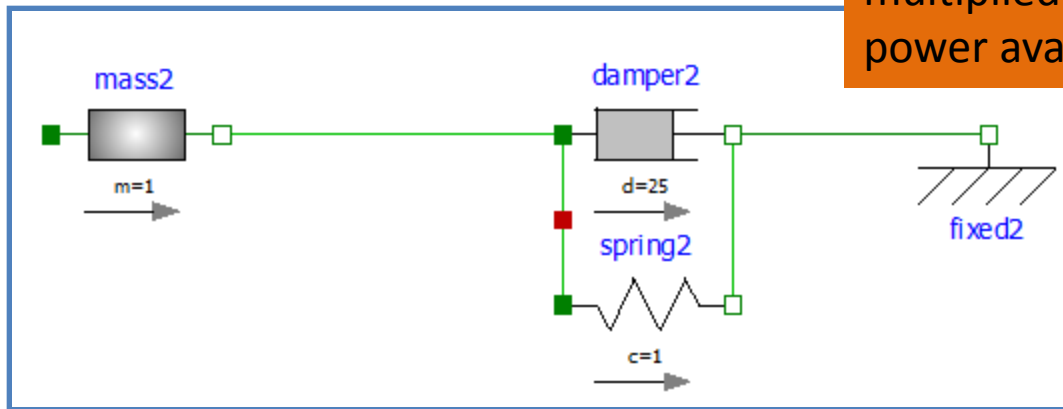


Reference: <https://www.modelica.org/ModelicaLibrariesOverview#mechanics>

Physical-Interaction Modeling Example: Mass-Spring-Damper System

Components represent physical entities interacting with other components through ports.

Ports are composed of through and across variables which multiplied together describe the power available at the port.



Connections represent **ACAUSAL physical interactions** between components. Connections indicate the exchange of power between ports of equivalent type. Equations are derived for each connection based on the concept of power continuity.

Modeling Physical Systems

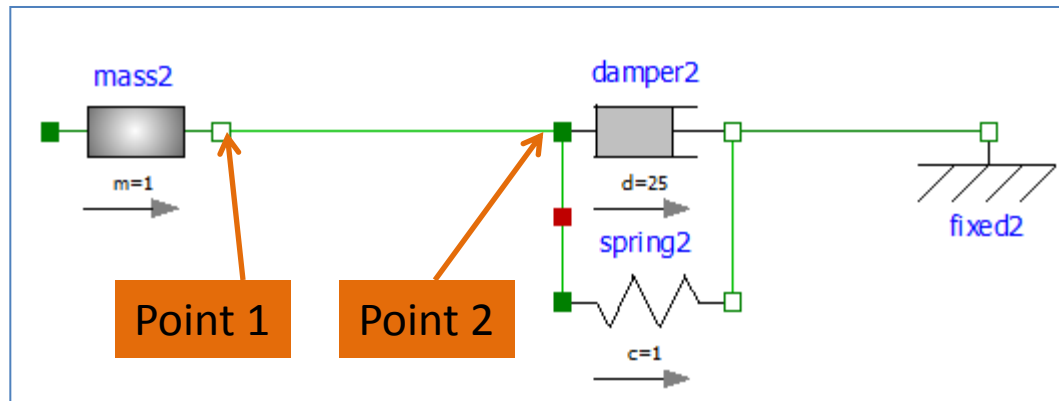
- Based on bond graph theory from the 60's
- Power continuity equations are formulated instead of energy conservation laws - This allows to connect components in a modular fashion
- Power continuity equations hold at the interfaces between components
- Power continuity equation: Sum of incoming power (energy flow) must be equal the sum of outgoing power

Modeling Physical Systems

Quick Intro (1)

Power Continuity Equation:

Power going into point 1 is equal to the power going out of point 2



$$P_{Mass2 @ P1} = P_{Damper2 @ P2} + P_{Spring2 @ P2}$$

Modeling Physical Systems

Quick Intro (2)

As Mechanical Power = Force * Speed, power continuity equation between point 1 and 2 can be formulated as :

$$F_{Mass2@P1} * v_{Mass2@P1} = F_{Damper2@P2} * v_{Damper2@P2} + F_{Spring2@P2} * v_{Spring2@P2}$$

As the mass is connected to the damper and the spring, this **equality equation** can be derived: speed at point 1 = speed at point 2 (speed is an **across variable**)

$$v_{Mass2@P1} = v_{Damper2@P2} = v_{Spring2@P2}$$

Based on the equality equation, the power continuity equation can be expressed as a **sum-to-zero equation** with the force as a **through variable**

$$F_{Mass2@P1} = F_{Damper2@P2} + F_{Spring2@P2}$$

By expressing the force in terms of the position, the **equation of motion** can be derived for the mass-spring-damper system

$$m_{Mass2} \ddot{x} = -k_{Damper2} \dot{x} - c_{Spring2} x$$

Functional Mock-up Unit

- Blocks performing computations based on state space representation can be defined according to the Functional Mock-up Interface (<https://www.fmi-standard.org/>)

Note: No concept of ports in FMU but block inputs and outputs can be described through ports

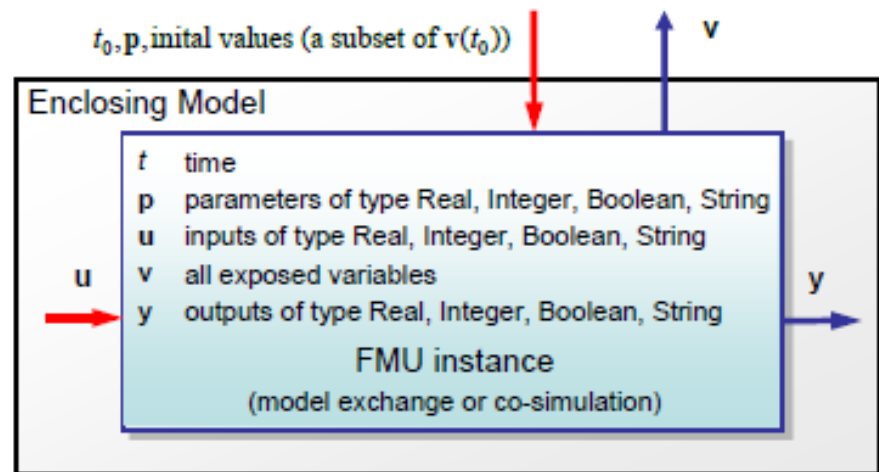


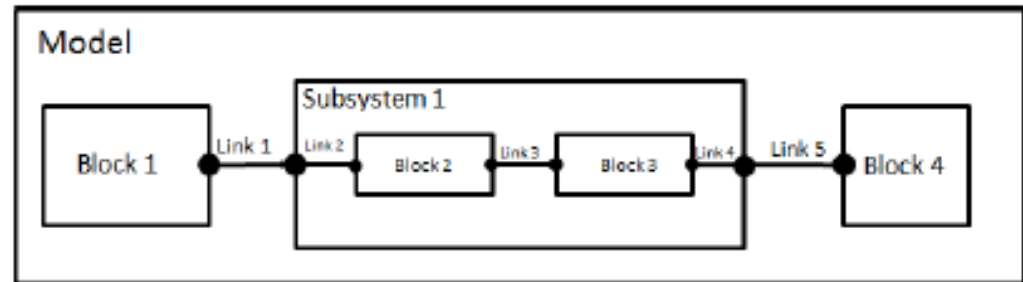
Figure 1: Data flow between the environment and an FMU. For details, see chapters 3 and 4.

Blue arrows: Information provided by the FMU.

Red arrows: Information provided to the FMU.

Common Modeling Constructs for describing Dynamic Systems

- Models
- Subsystems
- Blocks
- Ports
- Connections



Reference: „An Analysis of Solver-Based Simulation Tools“,
NIST Interagency/Internal Report (NISTIR) 7846, Published: 3/7/2012,
Ion Matei, Conrad E Bock, [http://www.nist.gov/manuscript-publication-search.cfm?
pub_id=909924](http://www.nist.gov/manuscript-publication-search.cfm?pub_id=909924)

Differences between Tools for Modeling Dynamic Systems

- Definition of blocks instances/usages: Explicit (e.g. Simulink) vs. implicit (e.g. Modelica) instantiation of block types/definitions
- Different possibilities to define blocks: With/without specialization of other blocks and with/without redefinition of properties of other blocks
- Definition of state equations: declarative (e.g. Modelica) vs. imperative form (e.g. Simulink, FMI)
- Interpretation of variable types (e.g. Real, double, float, etc...)
- Definition of initial conditions: as regular equation or as condition associated with a state variable
- Syntax for defining equations: Modelica supports the definition of DAEs with the *der* function meaning the derivative of a variable. Matlab/Simulink supports the definition of DAEs with functions describing the right side of differential equations

Metamodel for Common Modeling Constructs

Metamodel defined in Ecore can be easily modified and extended!

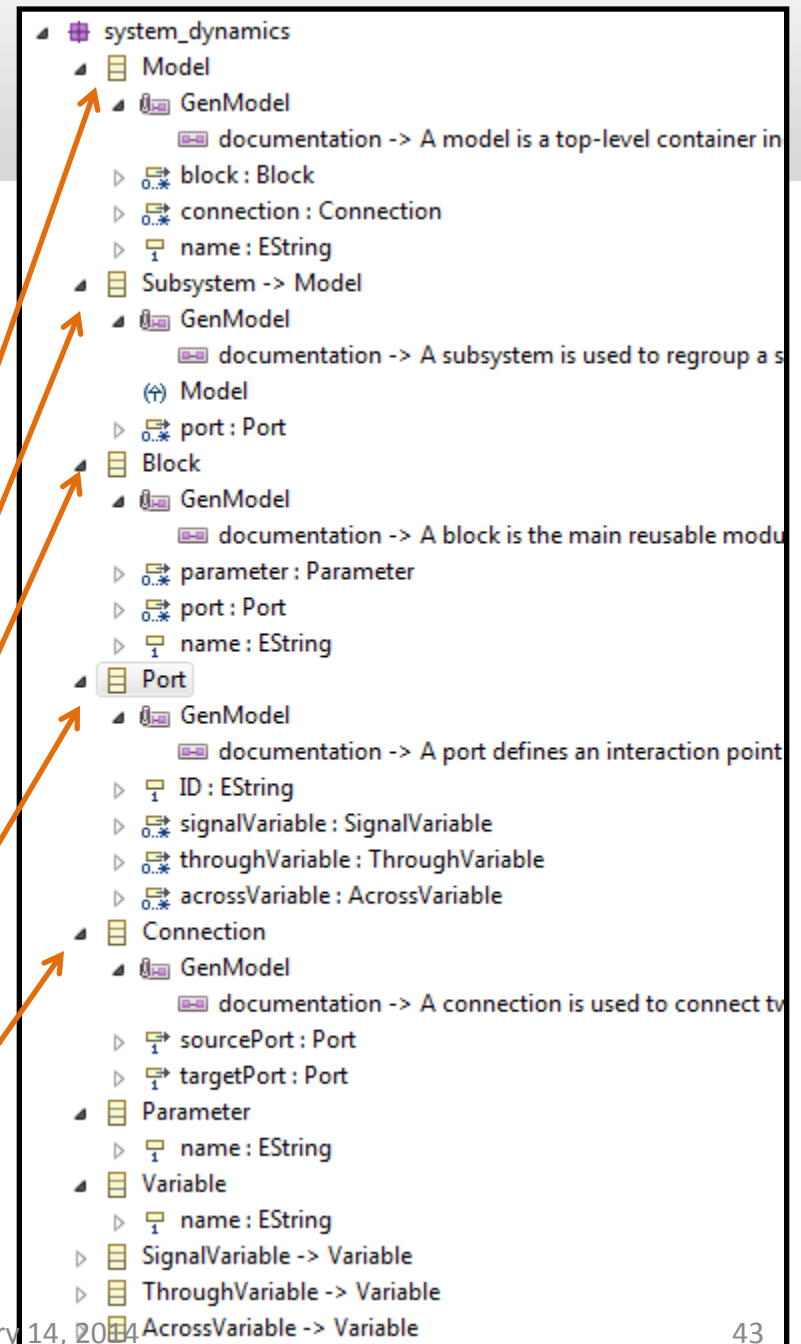
Models have blocks and connections

Subsystems are models with ports





































Blocks have parameters and ports

Ports have signal, and/or through and across variables

Connections have a source and a target port

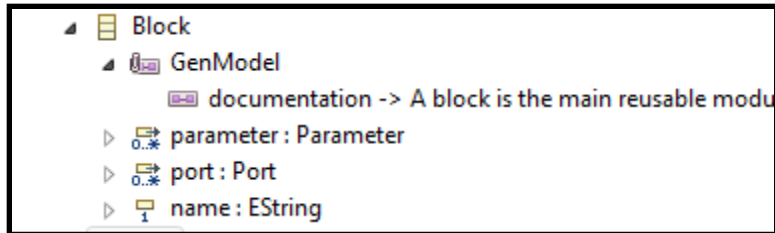


Part III: Creation of OSLC Specification for Dynamic Simulation Models

- ▲  system_dynamics
 - ▲  Model
 - ▲  GenModel
 -  documentation -> A model is a top-level container in
 - ▷  block : Block
 - ▷  connection : Connection
 - ▷  name : EString
 - ▲  Subsystem -> Model
 - ▲  GenModel
 -  documentation -> A subsystem is used to regroup a s
 - (⇄) Model
 - ▷  port : Port
 - ▲  Block
 - ▲  GenModel
 -  documentation -> A block is the main reusable modu
 - ▷  parameter : Parameter
 - ▷  port : Port
 - ▷  name : EString
 - ▲  Port
 - ▲  GenModel
 -  documentation -> A port defines an interaction point
 - ▷  ID : EString
 - ▷  signalVariable : SignalVariable
 - ▷  throughVariable : ThroughVariable
 - ▷  acrossVariable : AcrossVariable
 - ▲  Connection
 - ▲  GenModel
 -  documentation -> A connection is used to connect tv
 - ▷  sourcePort : Port
 - ▷  targetPort : Port
 - ▲  Parameter
 - ▷  name : EString
 - ▲  Variable
 - ▷  name : EString
 - ▷  SignalVariable -> Variable
 - ▷  ThroughVariable -> Variable
 - ▷  AcrossVariable -> Variable

Conversion of Ecore Metamodel to RDF Resources of corresponding OSLC Specification

Conversion of Ecore Metaclass and Features into RDFS Class and RDF properties



Reuse of
Dublin Core
Vocabulary

Namespace URI
of RDF Vocabulary

```
<rdf:Class rdf:about="dynsim:Block">
  <rdf:label xml:lang="en-GB">Block</rdf:label>
  <dcterms:description xml:lang="en-GB">A block is the main reusable modular unit
  for describing a physical system or an operation within a dynamic system model.
  </dcterms:description>
  <rdf:isDefinedBy rdf:resource="http://incose.org/dynsim#" />
  <dcterms:issued>2014-01-05</dcterms:issued>
</rdf:Class>
<rdf:Property rdf:about="dynsim:Block_name">
  <rdf:label xml:lang="en-GB">name</rdf:label>
  <rdf:isDefinedBy rdf:resource="http://incose.org/dynsim#" />
  <dcterms:issued>2014-01-05</dcterms:issued>
</rdf:Property>
<rdf:Property rdf:about="dynsim:Block_parameter">
  <rdf:label xml:lang="en-GB">parameter</rdf:label>
  <rdf:isDefinedBy rdf:resource="http://incose.org/dynsim#" />
  <dcterms:issued>2014-01-05</dcterms:issued>
</rdf:Property>
<rdf:Property rdf:about="dynsim:Block_port">
  <rdf:label xml:lang="en-GB">port</rdf:label>
  <rdf:isDefinedBy rdf:resource="http://incose.org/dynsim#" />
  <dcterms:issued>2014-01-05</dcterms:issued>
</rdf:Property>
```

RDF Vocabulary for Dynamic Simulation Domain

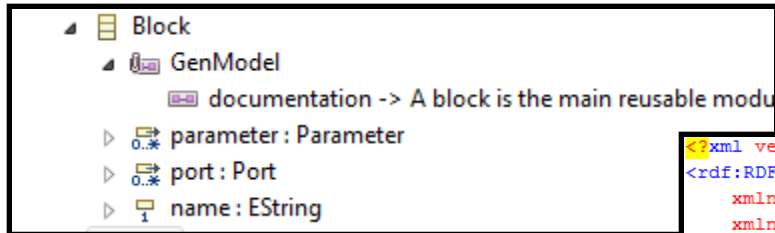
RDF Vocabulary
Namespace
prefix and URI

RDFS Classes
and RDF
Properties are
defined within
an RDF
Vocabulary

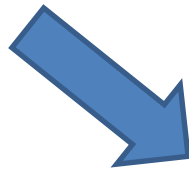
```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:dcterms="http://purl.org/dc/terms/"
  xmlns:dynsim="http://incose.org/dynsim#"
  <rdfs:Class rdf:about="dynsim:Model">
    <rdfs:label xml:lang="en-GB">Model</rdfs:label>
    <dcterms:description xml:lang="en-GB">A model is a top-level contain
    <rdfs:isDefinedBy rdf:resource="http://incose.org/dynsim#">
    <dcterms:issued>2014-01-05</dcterms:issued>
  </rdfs:Class>
  <rdf:Property rdf:about="dynsim:Model_name">
    <rdfs:label xml:lang="en-GB">name</rdfs:label>
    <rdfs:isDefinedBy rdf:resource="http://incose.org/dynsim#">
    <dcterms:issued>2014-01-05</dcterms:issued>
  </rdf:Property>
  <rdf:Property rdf:about="dynsim:Model_block">
    <rdfs:label xml:lang="en-GB">block</rdfs:label>
    <rdfs:isDefinedBy rdf:resource="http://incose.org/dynsim#">
    <dcterms:issued>2014-01-05</dcterms:issued>
  </rdf:Property>
  <rdf:Property rdf:about="dynsim:Model_connection">
    <rdfs:label xml:lang="en-GB">connection</rdfs:label>
    <rdfs:isDefinedBy rdf:resource="http://incose.org/dynsim#">
    <dcterms:issued>2014-01-05</dcterms:issued>
  </rdf:Property>
  <rdfs:Class rdf:about="dynsim:Subsystem">
    <rdfs:label xml:lang="en-GB">Subsystem</rdfs:label>
    <dcterms:description xml:lang="en-GB">A subsystem is used to regroup
    <rdfs:isDefinedBy rdf:resource="http://incose.org/dynsim#">
    <dcterms:issued>2014-01-05</dcterms:issued>
    <rdfs:subClassOf rdf:resource="dynsim:Model"/>
  </rdfs:Class>
```

Just a snippet...

Conversion of Ecore Metaclass and Features into OSLC Resource Shape



Property names and multiplicities are converted into OSLC resource shape constraints



```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:oslc="http://open-services.net/ns/core#"
  xmlns:dcterms="http://purl.org/dc/terms/"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:dynsim="http://incose.org/dynsim#"
  <oslc:ResourceShape rdf:about="http://myOSLCServiceProvider.com/dynsim/BlockResourceShape">
    <oslc:describes rdf:resource="dynsim:Block"/>
    <dcterms:title rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral">
      Block Resource Shape</dcterms:title>
    <oslc:property>
      <oslc:Property>
        <oslc:name>name</oslc:name>
        <oslc:propertyDefinition rdf:resource="dynsim:Block_name"/>
        <oslc:occurs rdf:resource="http://open-service.net/ns/core#Exactly-one"/>
      </oslc:Property>
    </oslc:property>
    <oslc:property>
      <oslc:Property>
        <oslc:name>parameter</oslc:name>
        <oslc:propertyDefinition rdf:resource="dynsim:Block_parameter"/>
        <oslc:range rdf:resource="dynsim:Parameter"/>
        <oslc:valueType rdf:resource="http://open-services.net/ns/core#Resource"/>
        <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      </oslc:Property>
    </oslc:property>
    <oslc:property>
      <oslc:Property>
        <oslc:name>port</oslc:name>
        <oslc:propertyDefinition rdf:resource="dynsim:Block_port"/>
        <oslc:range rdf:resource="dynsim:Port"/>
        <oslc:valueType rdf:resource="http://open-services.net/ns/core#Resource"/>
        <oslc:occurs rdf:resource="http://open-service.net/ns/core#Zero-or-many"/>
      </oslc:Property>
    </oslc:property>
  </oslc:ResourceShape>
</rdf:RDF>
```