SIEMENS

Configuration Workshop 2014 Novi Sad/Нови Сад

Integrating Distributed Configurations with RDFS and SPARQL

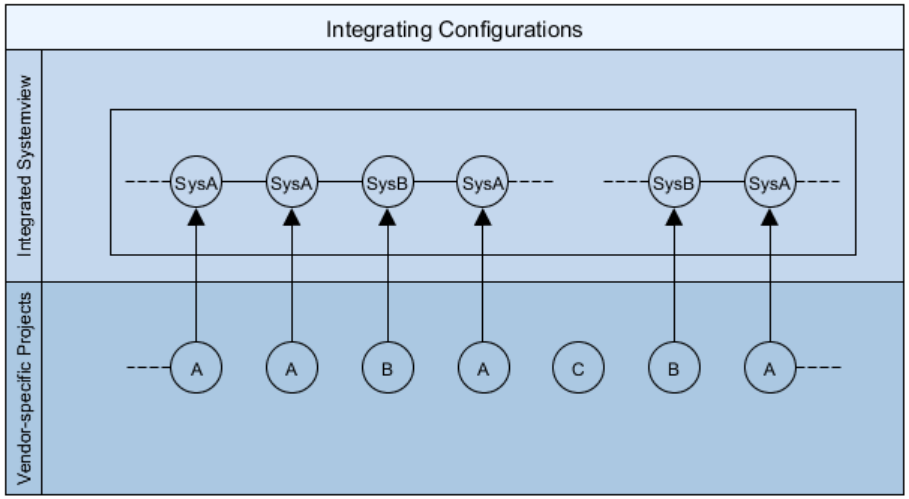
Gottfried Schenner, Stefan Bischof, Axel Polleres, Simon Steyskal

Integrating Distributed Configurations with RDFS and SPARQL
Use Case

Large technical systems are sometimes configured with multiple product configurators
These configurators use their own ontologies
Knowledge engineers would rather share a toothbrush than share a concept name

Example Railway Line:

- Individual stations are configured by different vendors
- These configurators store their configurations in separate databases (projects)
- How can the infrastructure owner (i.e. railway company) write queries about the whole system in a vendor independent manner?



Integrating Distributed Configurations with RDFS and SPARQL Using Semantic Web Technologies

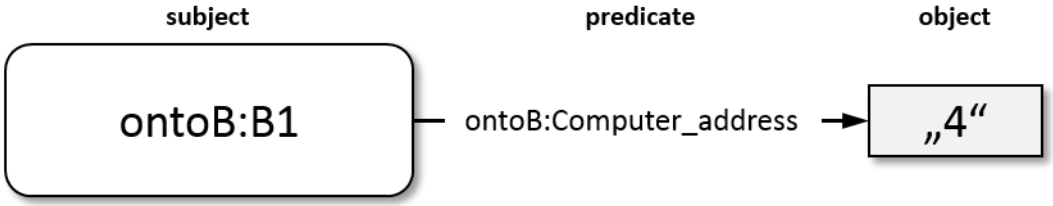
Looking for a “lightweight” non-proprietary approach using only standard Semantic Web technologies

Our approach uses:

- RDF(S) to represent configurations and knowledge bases (ontologies)
- SPARQL (standard RDF query language) to create linked system view
- A little bit of OWL

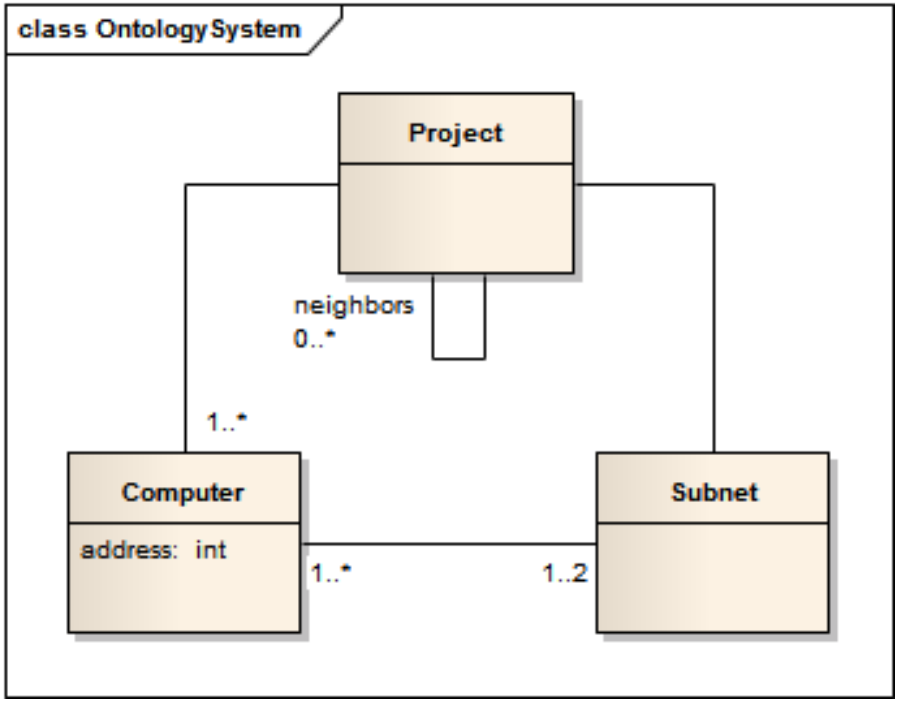
Example:

RDF-graph consists of triple: subject (URI) predicate(URI) object(URI or Literal)

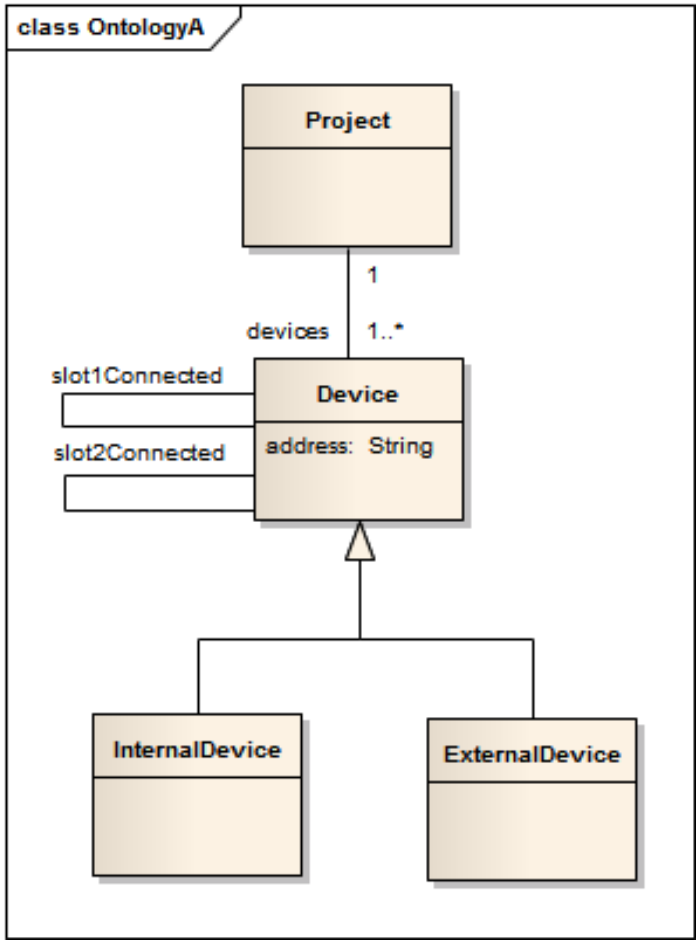


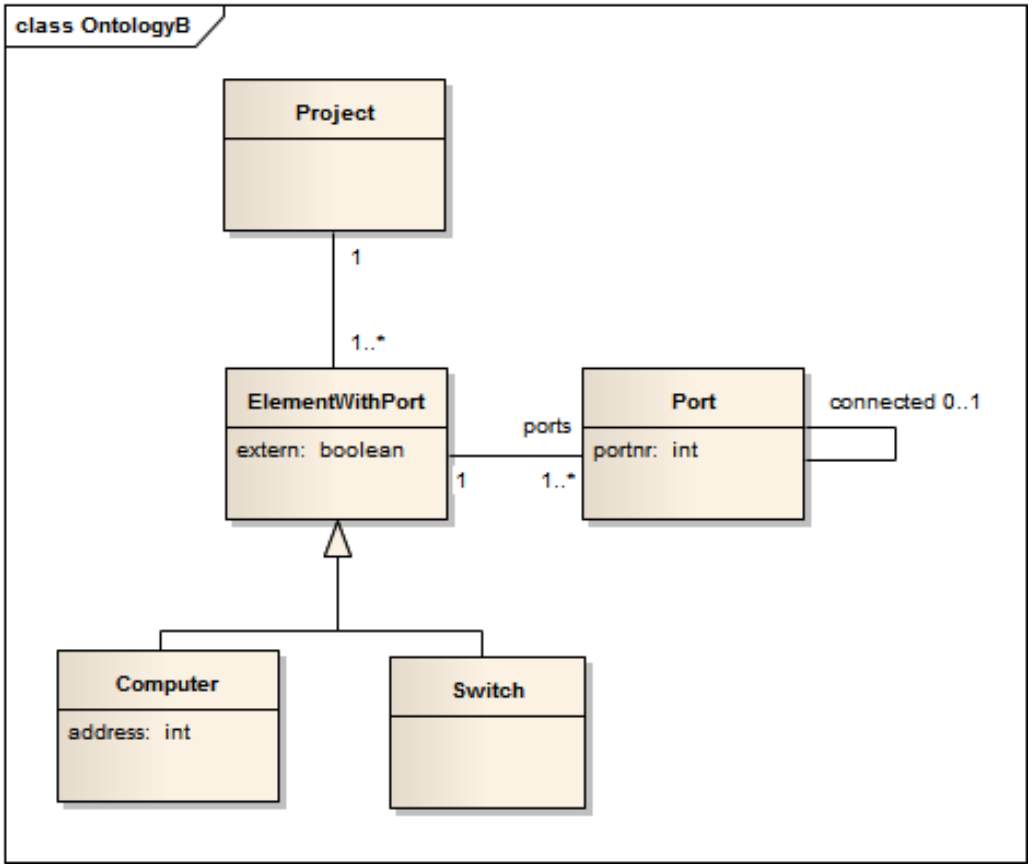
Integrating Distributed Configurations with RDFS and SPARQL
Working Example – System Ontology

- Fictitious computer network
  - Every computer has an unique address
  - A computer can be part of 1-2 subnets
  - A computer is part of exactly one project
  - A project is some arbitrary subdivision of the whole network

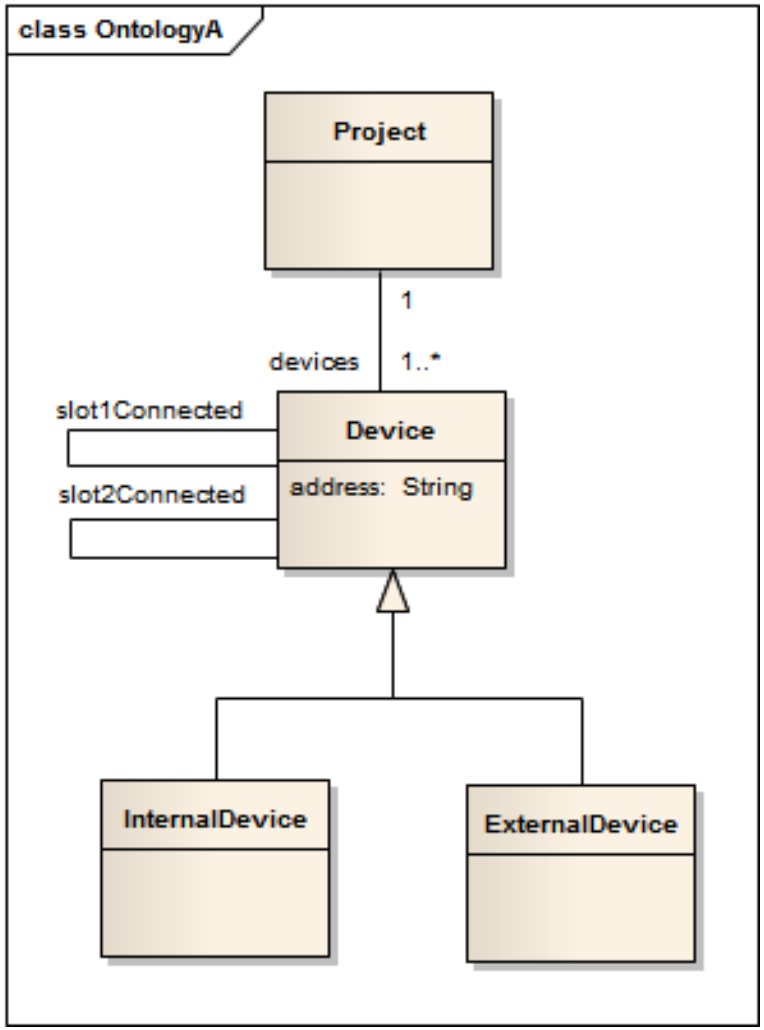


Integrating Distributed Configurations with RDFS and SPARQL
Working Example – Configurator-specific Ontologies





Integrating Distributed Configurations with RDFS and SPARQL
From UML to RDF




```
ontoA:InternalDevice
  rdf:type owl:Class ;
  rdfs:subClassOf ontoA:Device .

ontoA:Device_address
  rdf:type owl:DatatypeProperty;
  rdfs:domain ontoA:Device ;
  rdfs:range xsd:string .

ontoA:Device_slot1Connected
  rdf:type owl:ObjectProperty ;
  rdfs:range ontoA:Device ;
  rdfs:domain ontoA:Device .

# instance data
ontoA:A1
  rdf:type ontoA:InternalDevice ;
  ontoA:Device_address "1";
  ontoA:Device_slot1Connected
           ontoA:A2 , ontoA:A3 .
```

Integrating Distributed Configurations with RDFS and SPARQL
Open World Assumption & Unique Name Assumption

Semantic Web Technologies usually use Open World Assumption and don't use Unique Name Assumption

Open (OWA) vs. Closed World Assumption (CWA)

```
# How many times was Elvis married? (OWA)
:Elvis :isMarriedTo :Priscilla .
# How many connections does A1 have? (CWA)
:A1 :Device_slot1Connected :A2 , :A3 .
```

Unique Name Assumption (UNA)

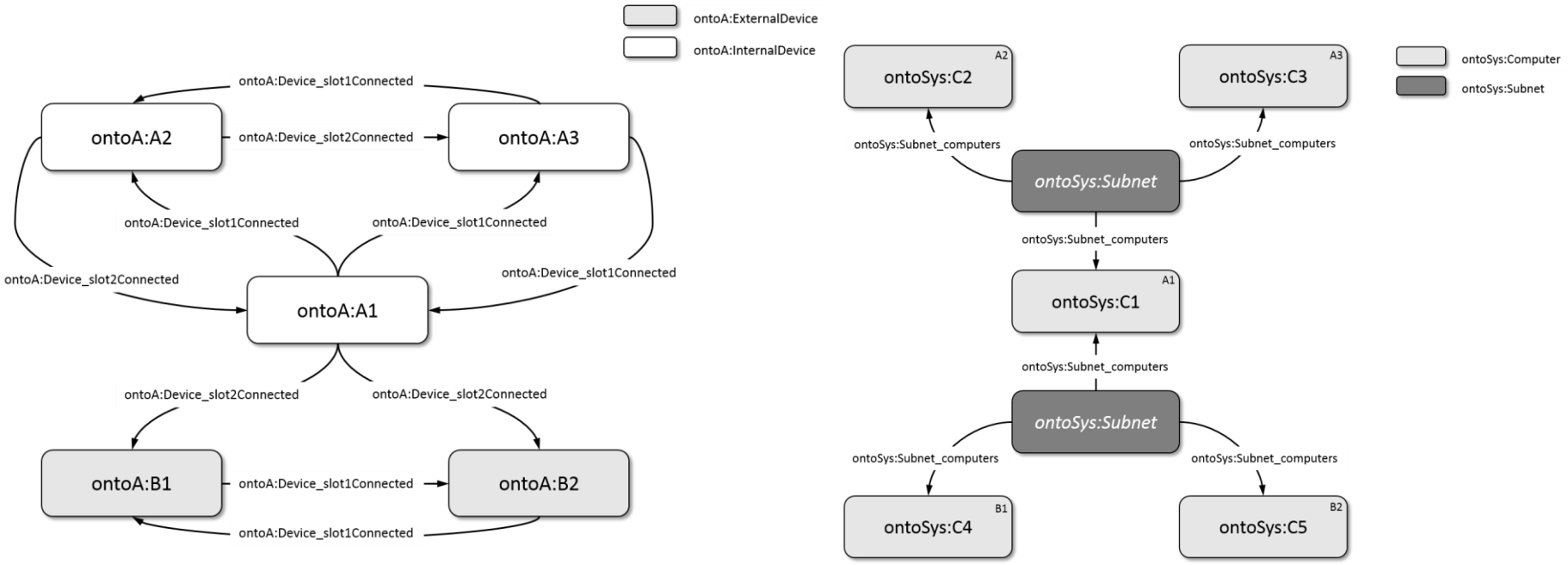
```
:TheKing :loves :PeanutbutterBananaSandwich . (Without UNA)
:Elvis owl:sameAs :TheKing .
:A1 :Device_slot1Connected :A2 , :A3 . (UNA)
```

Pragmatic solution: Use SPARQL (CWA) and use UNA by Default, i.e. treat identifiers as different objects unless stated otherwise with owl:sameAs

Integrating Distributed Configurations with RDFS and SPARQL
Creating System View

How do we get from instance data of ontology A (left) to System View (right)?

- Create System View from vendor-specific view with SPARQL CONSTRUCT queries



Integrating Distributed Configurations with RDFS and SPARQL
Creating Instances

New instances can be created either by creating a new URI or by using blank nodes.

Example new URI:

```
CONSTRUCT {
  ?computeruri rdf:type ontoSys:Computer.
  ?computeruri ontoSys:Computer_address ?address.
}
WHERE{
  ?device ontoA:Device_address ?address.
  BIND(URI(CONCAT(URISYS,STR(?address)))
      AS ?computeruri)
}
```

Example blank node:

```
CONSTRUCT {
    _:blanknodeForProject rdf:type ontoSys:Project .
    _:blanknodeForProject ontoSys:origin ?project .
}
WHERE{
    ?project rdf:type ontoA:Project .
}
```

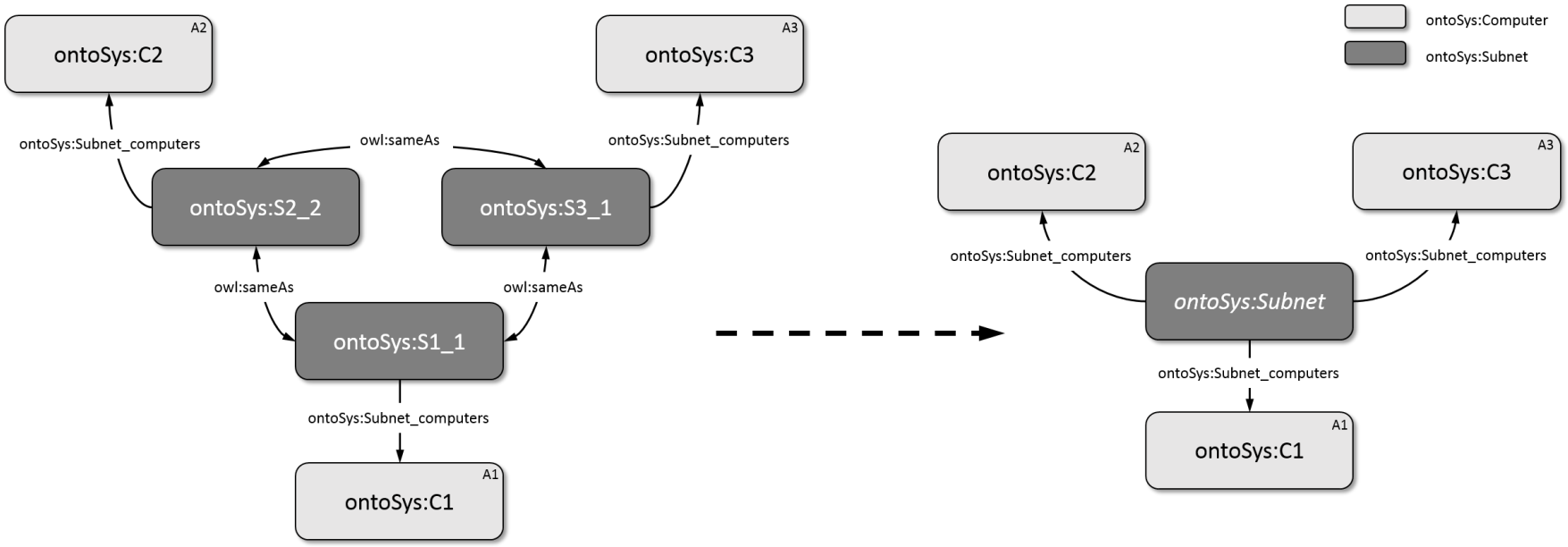
Integrating Distributed Configurations with RDFS and SPARQL Complex Mappings with owl:SameAs

Using multiple URIs for the same object makes complex mappings easier

Example Creating Subnets:

```
# Whenever 2 Devices are directly connected in OntologyA, they are
part of the same Subnet in OntologySys
CONSTRUCT {
  ?sub1 ontoSys:Subnet_computers ?c1 .
  ?sub2 ontoSys:Subnet_computers ?c2 .
  ?sub1 rdf:type ontoSys:Subnet .
  ?sub2 rdf:type ontoSys:Subnet .
  ?sub1 owl:sameAs ?sub2 .
} WHERE {
?d1 ontoA:Device_slot1Connected ?d2
…}
```

Integrating Distributed Configurations with RDFS and SPARQL
Complex Mappings with owl:SameAs (cont.)

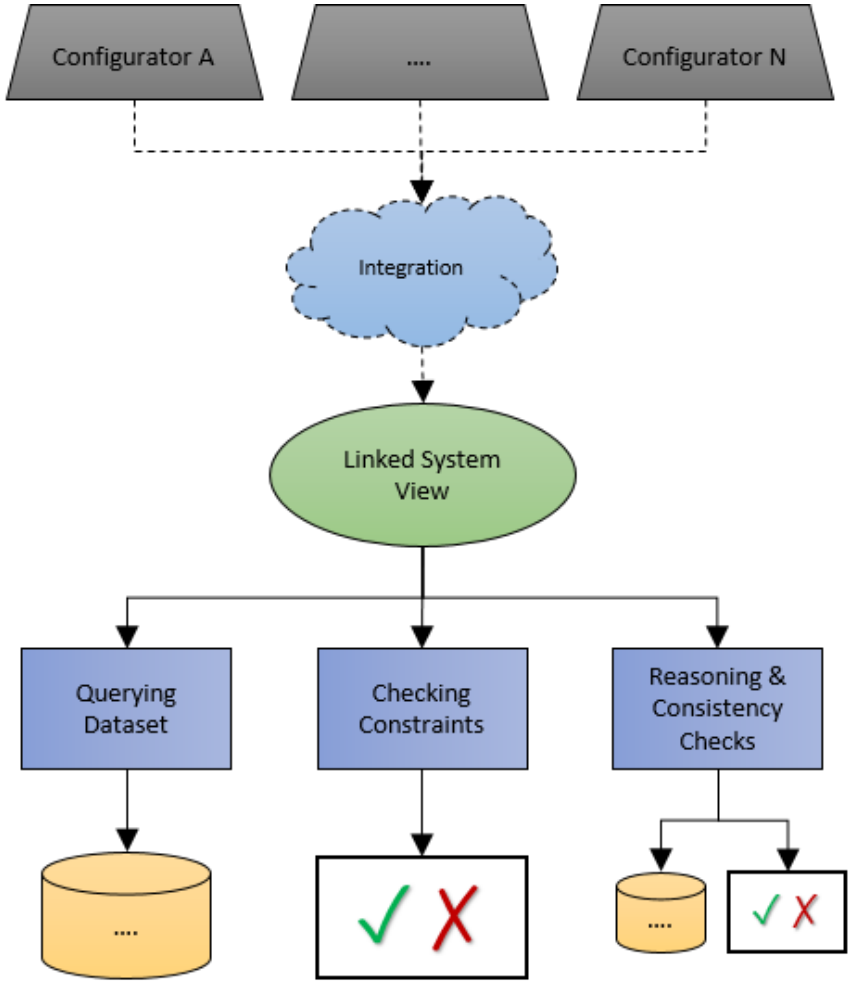


Integrating Distributed Configurations with RDFS and SPARQL
Result of Data Integration

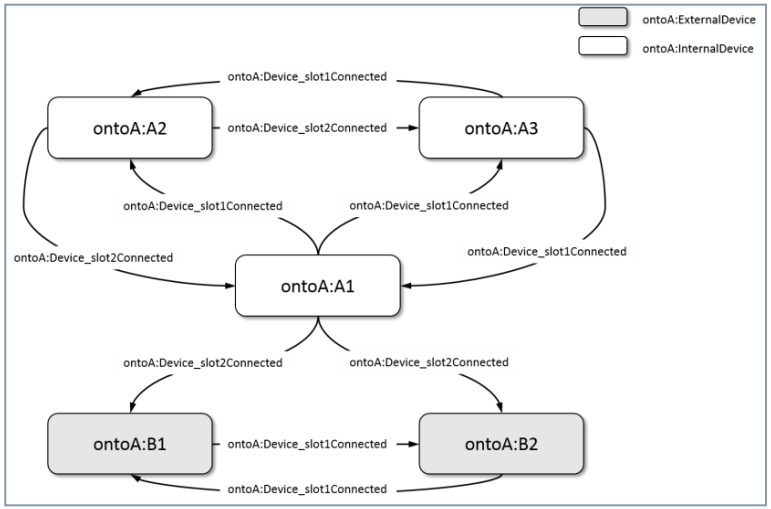
After the data integration we can query the system view, check consistency etc. with SPARQL queries

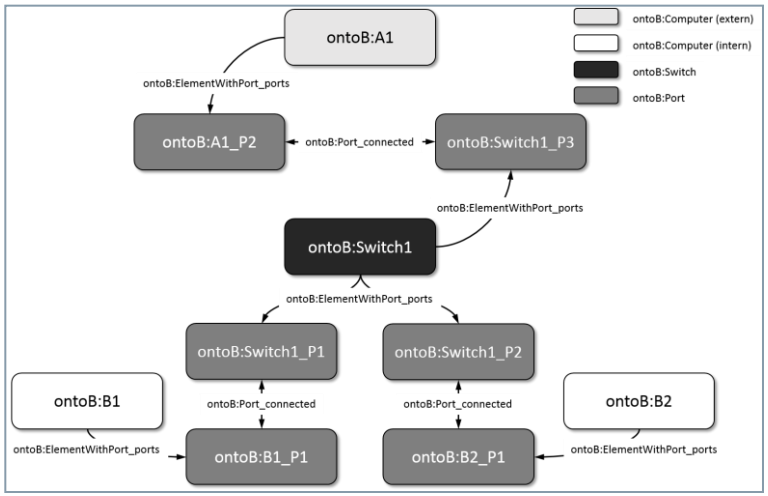
Example:

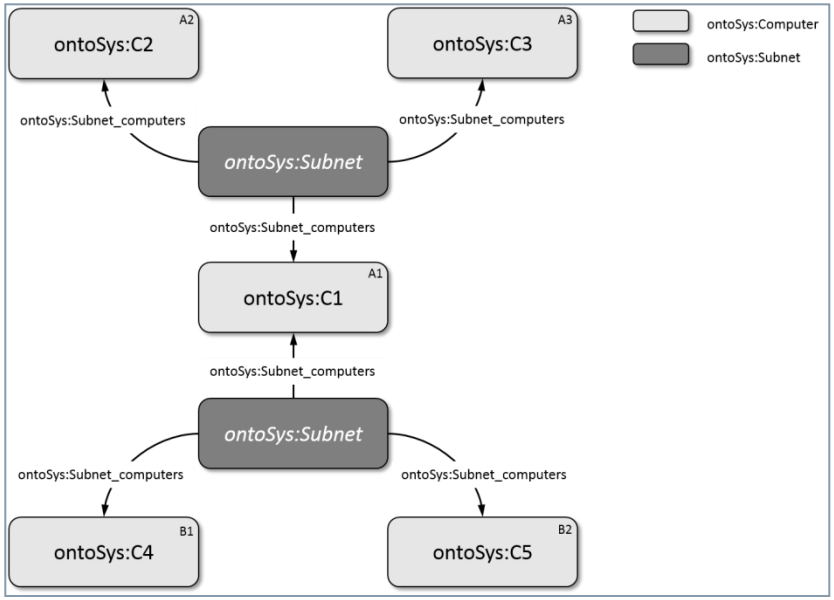
```
# return all the addresses used in all
projects
SELECT ?p ?address
WHERE {
 ?p ontoSys:Project_computers ?c .
 ?c ontoSys:Computer_address ?address .
}
```



Integrating Distributed Configurations with RDFS and SPARQL Recap







SIEMENS

How would you approach the Data Integration scenario?

Semantic Web Technologies and Product Configuration?

How do we share Product Configuration Knowledge Bases?

SIEMENS

Thank you for your attention!

Gottfried Schenner
Corporate Technology
Research Group
Configuration Technologies

Siemensstraße 90
1210 Vienna

Phone: +43 (0) 51707 35419

E-mail:
gottfried.schenner@siemens.com