Semantic Interoperability in IT Security: Ontology for IT Product Representation

Paul Cichonski
Booz Allen Hamilton
National Institute of Standards and Technology (NIST)

Introduction (1 of 2)

There are many challenges to implementing information system security (ISS) measures for private and public organizations:

- Number and variety of systems to secure
- Need to comply with mandates
- Need to respond quickly to new threats
- Need for interoperability across disparate organizations and agencies (e.g. across entire Federal Government infrastructure)

Introduction – (2 of 2)

"NIST is a non-regulatory federal agency within the U.S. Department of Commerce. ***NIST's mission*** *is to promote U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve our quality of life."*

- *http://www.nist.gov/public_affairs/general_information.cfm*

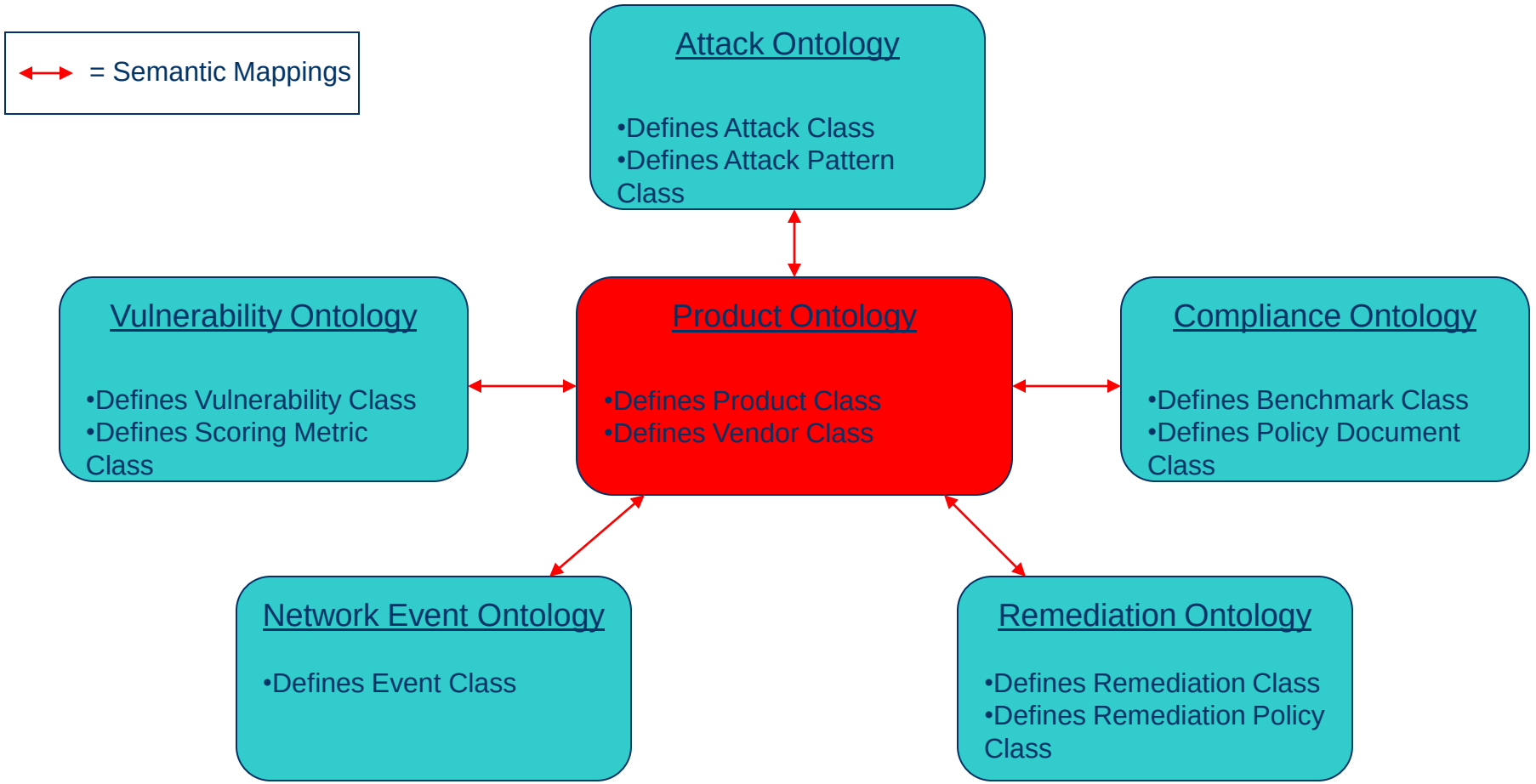
Agenda

- What is the NIST Security Automation Program?
- What is the value of standardized IT security data models?
- How do IT product ontologies fit in?
- How does this work compare with other product ontologies?
- What does the current ontology look like?

NIST Security Automation Program is all about interoperability

- Program designed to create standardized communication and reporting data models around IT security.
  - Multiple domains to model including compliance, vulnerabilities, events, remediation, and reporting.
  - Goal is vertical and horizontal interoperability
- Focus is on increasing the level of interoperability between heterogeneous IT security domains.
  - Enables fast and accurate correlation within the enterprise and across organizations/agencies.
  - Interoperability will allow diverse tool suites and repositories to share data across multiple security domains.

Product Data is Central to IT Security Data Models



Important Definitions

- **Product** – "*complete set of computer programs, procedures and associated documentation and data designed for delivery to a software consumer*"
  - Definition from ISO 19770-2 Standard
  - Represents the product model, not a physical instantiation of a product.

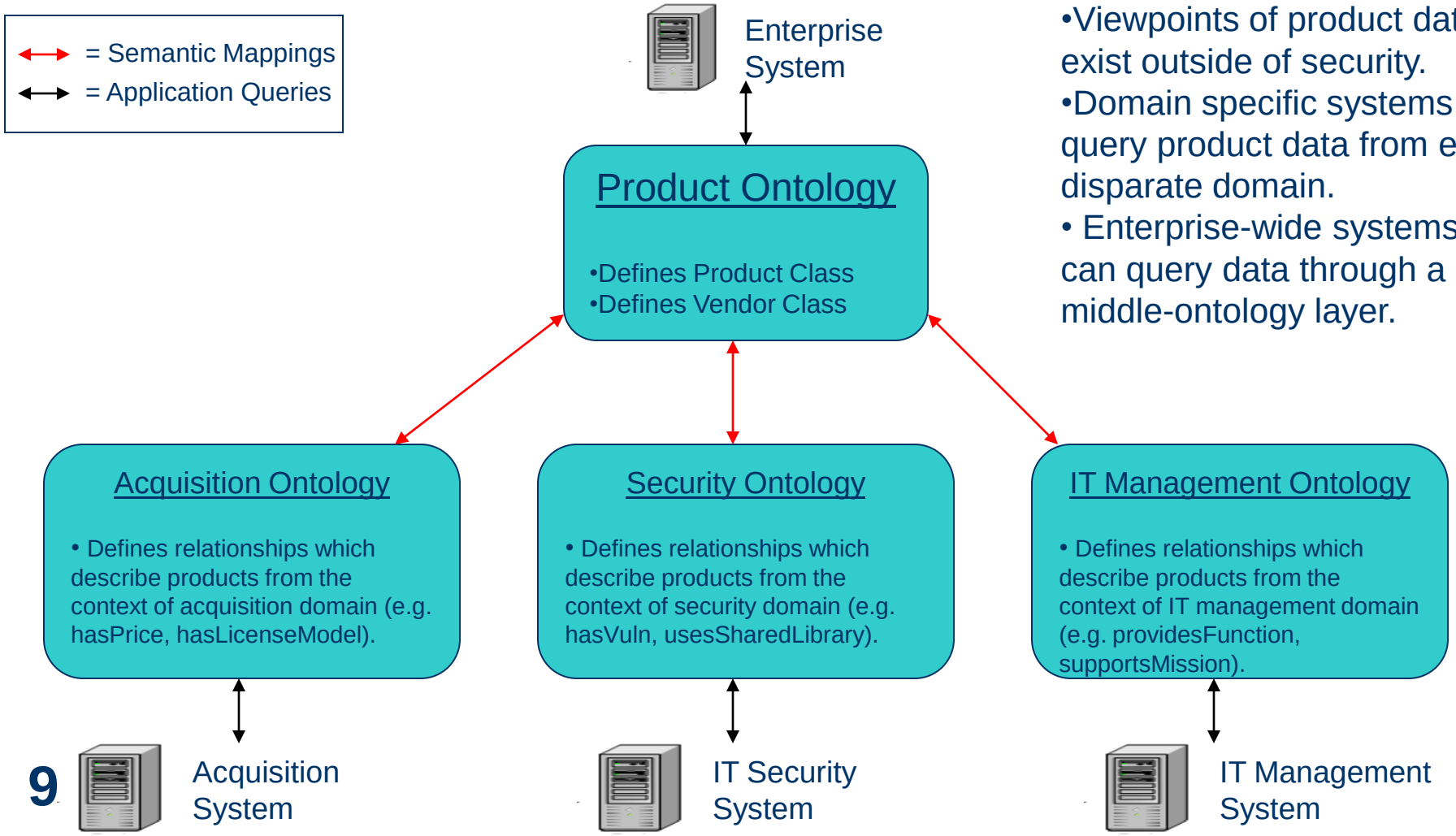
Why do we need another product ontology?

- Don't we have enough?

ISO 19770-2

- Not yet, no model is focusing on the security viewpoint of product data.
  - Or at least I haven't found it.
- Semantic links are possible between models.

Multiple viewpoints of product data exist and separate ontologies are needed to model the disparate viewpoints



- Viewpoints of product data exist outside of security.
- Domain specific systems can query product data from each disparate domain.
- Enterprise-wide systems can query data through a middle-ontology layer.

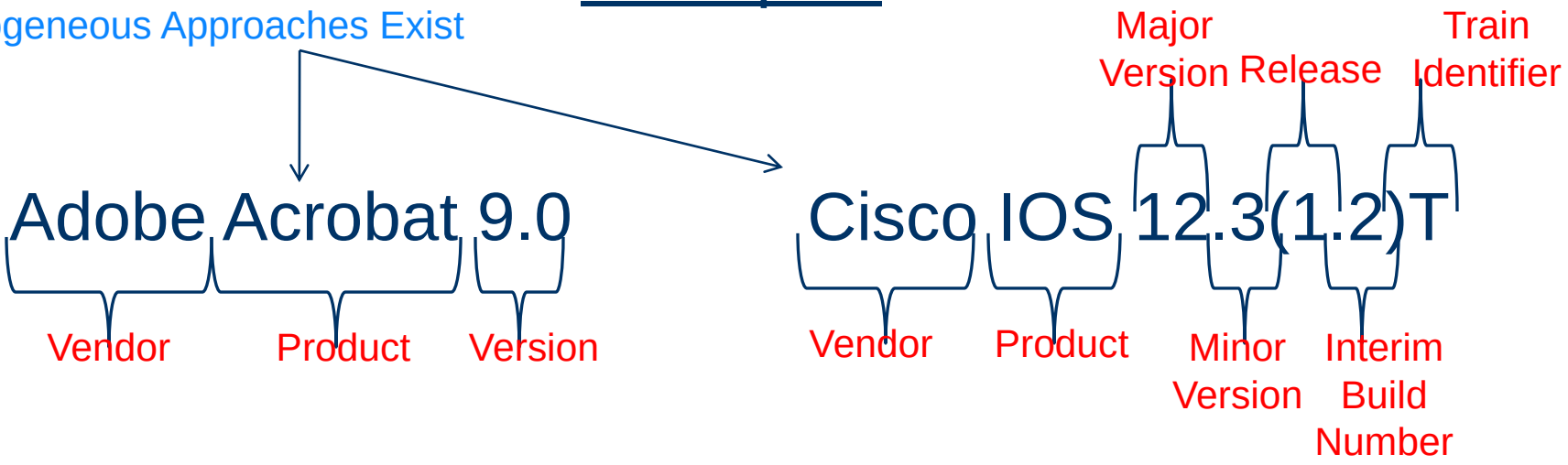
Overview of Product Ontology for National Vulnerability Database

Important Definitions

- **Identification Strategy** – The way in which an organization names and versions a product.

Examples

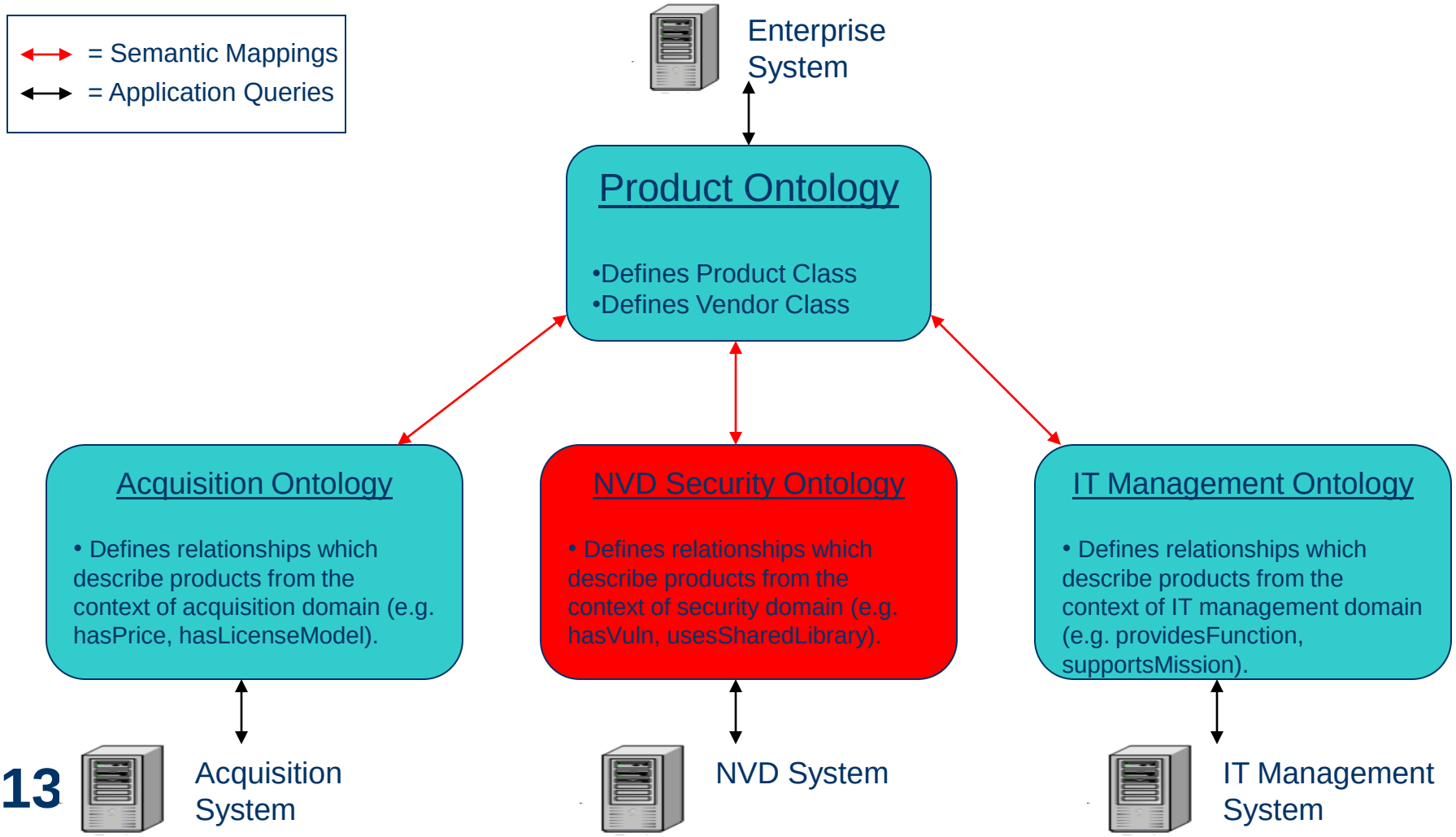
Heterogeneous Approaches Exist



What is the National Vulnerability Database (NVD)?

- US Government central repository of security automation content.
  - Holds vulnerability and configuration management XML data adhering to Security Automation Schemas.
  - Over 40,000 vulnerabilities, and 137 Security Checklists.
- Contains explicit links between vulnerability/configuration data and IT product data.
  - Referred to as applicability statements.
- NVD Product ontology is designed to capture and facilitate the relationships required within NVD.
  - Goal is to make the security data more meaningful.

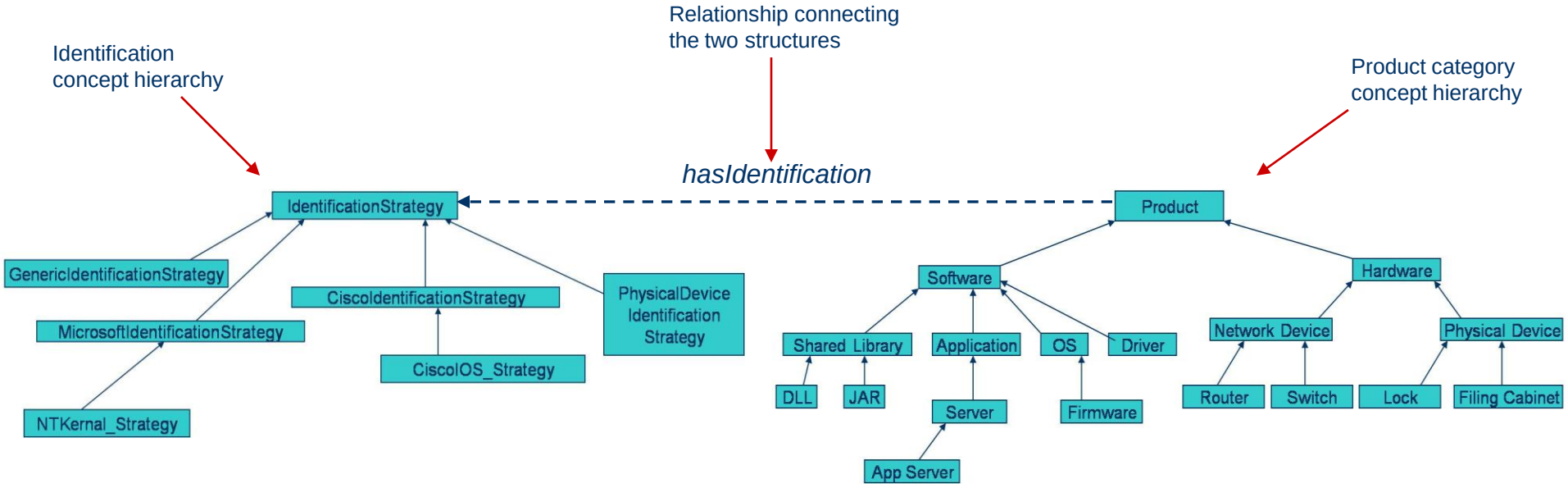
NVD Product Ontology begins to define IT security viewpoint of product data.



NVD Product Ontology Goals

- Ontology must support NVD's primary use case involving making statements of applicability between IT concepts (e.g. Vulnerabilities, Security Configuration Checklists) and IT products.
- Ontology must support the ability to make statements of applicability at various levels of abstraction and across ranges of products (e.g. Microsoft Windows version 4.3 to 5.6).
- Ontology must support the ability to capture granular product identification data which may vary on a per product basis.

High-level NVD Ontology Overview



= <owl:Class>

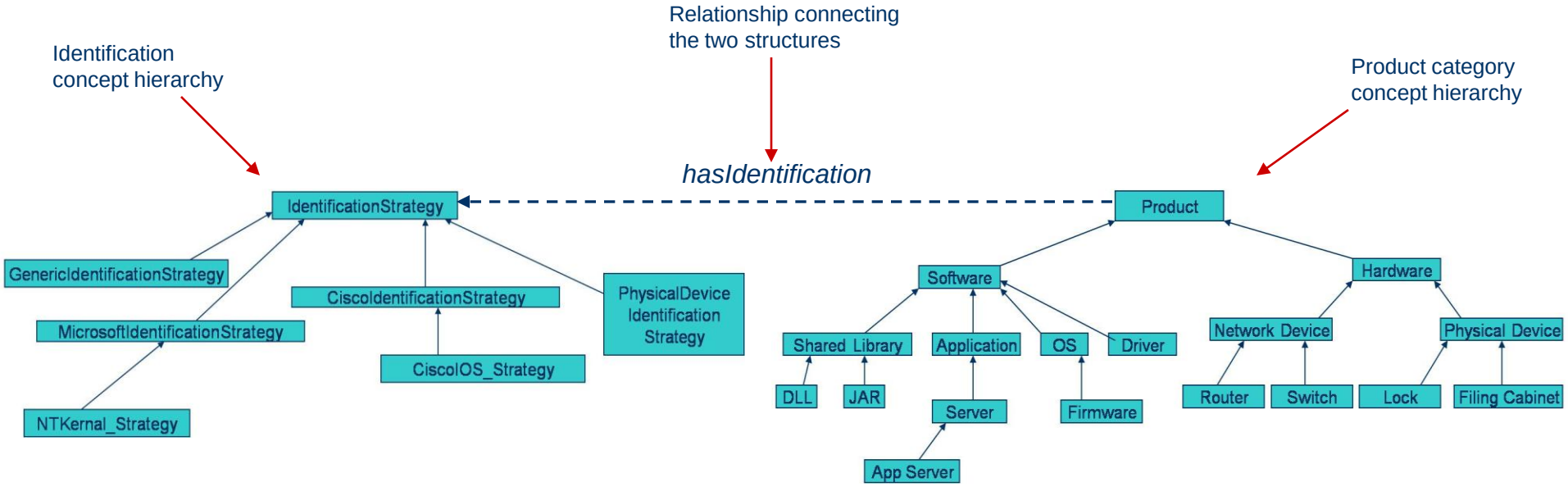
= <rdfs:subClassOf>

ABC = <rdf:Property>

Structure of the Ontology

- NVD Ontology models two separate concept structures as formal “is-a” hierarchies.
  - Category concept hierarchy
  - Identification concept hierarchy
- NVD Ontology also includes other types of semantic relationships.
  - Relationships between applications and codebases (“made up of” relationships)
  - Explicit differences between sets of products created by defining disjoint sets (e.g. hardware vs. software products)

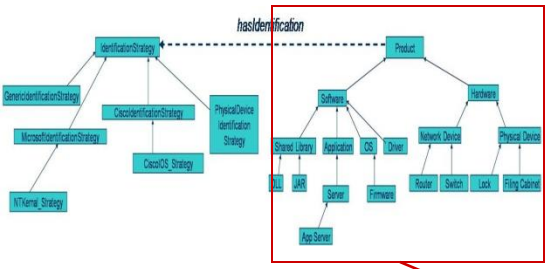
High-level NVD Ontology Overview



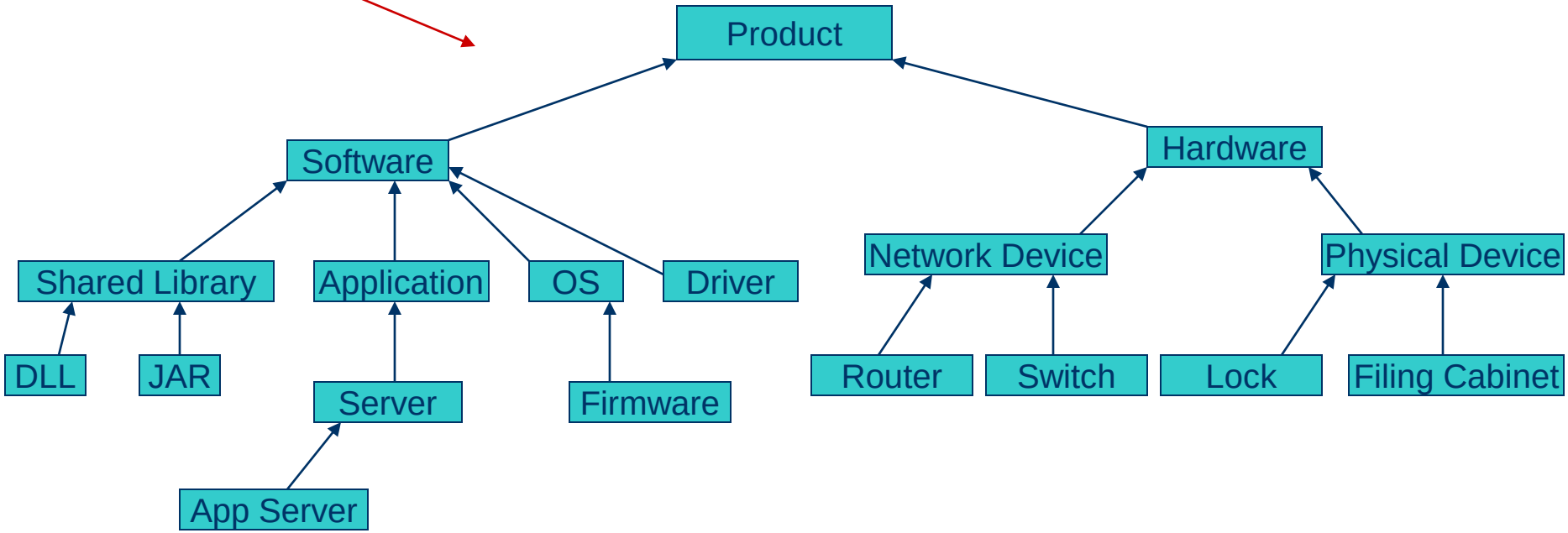
= <owl:Class>

= <rdfs:subClassOf>

ABC = <rdf:Property>



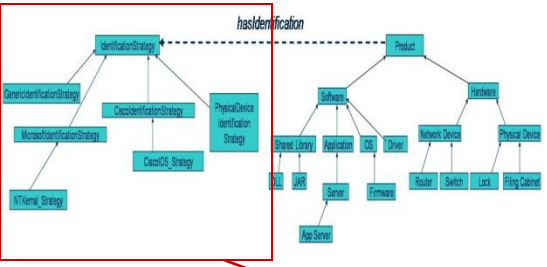
Product Category Hierarchy



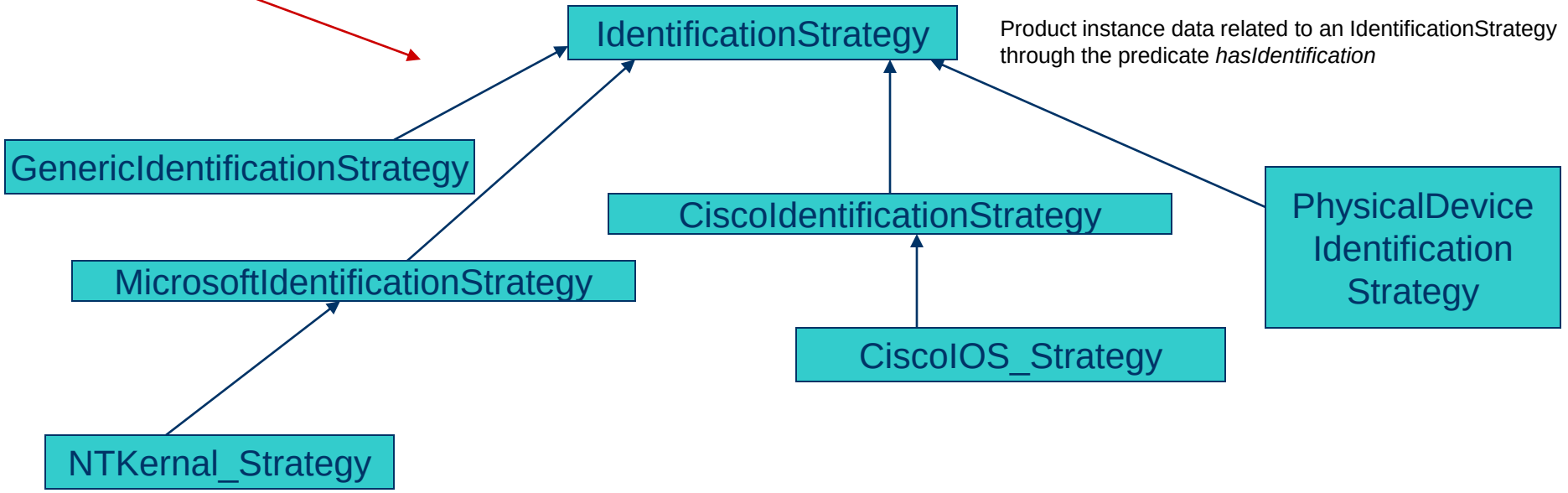
Possible Predicates

- *hasIdentification*, domain of Product, range of IdentificationStrategy, <owl:inverseFunctionalProperty>
- *hasReleaseDate*, domain of Product
- *hasCpeName*, domain of Product
- *usesSharedLibrary*, domain of Application, range of SharedLibrary
- *contains*, domain of Product, range of Product, inverseOf *containedIn*
- *hasOwner*, domain of Product, range of Foaf:Agent, inverseOf *ownedBy*
- *hasAutomationTest*, domain of Product
- Many other possibilities exist, very granular predicates can be defined further down tree.

= <owl:Class>
= <rdfs:subClassOf>
ABC = <rdf:Property>



Identification Concept Hierarchy



Product instance data related to an IdentificationStrategy through the predicate *hasIdentification*

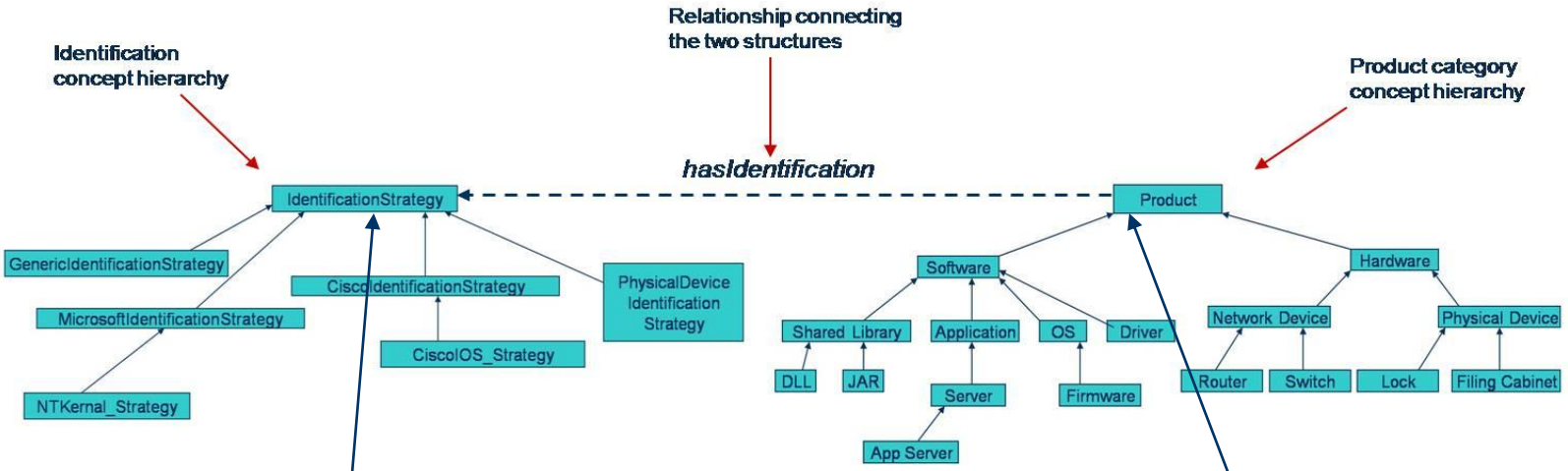
Possible Predicates

- *hasName*, domain of IdentificationStrategy
- *hasModelNumber*, domain of PhysicalDeviceIdentificationStrategy
- *hasCiscoTrainIdentifier*, domain of CiscoIOS_Strategy
- *hasCiscoInterimBuildNumber*, domain of CiscoIOS_Strategy
- *hasMicrosoftMajorVersion*, domain of NTKernal_Strategy
- *hasVersion*, domain of GenericIdentificationStrategy
- *hasUpdate*, domain of GenericIdentificationStrategy

= <owl:Class>
= <rdfs:subClassOf>
ABC = <rdf:Property>

Product Instance Data Instantiated from Model Classes

Model



Instance Data

Blank Node

blank_32320

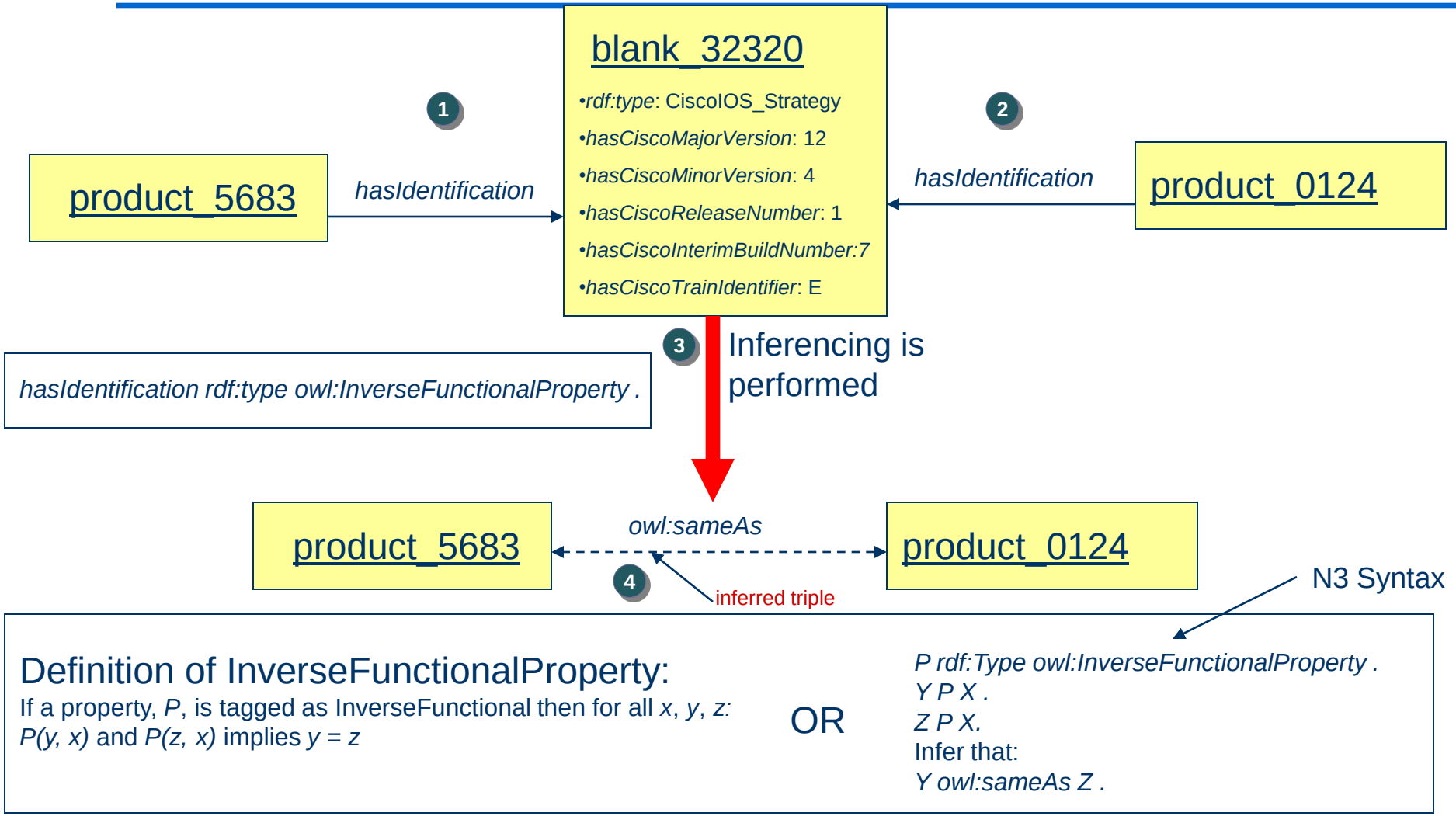
- *rdf:type*: CiscoIOS_Strategy
- *hasCiscoMajorVersion*: 12
- *hasCiscoMinorVersion*: 4
- *hasCiscoReleaseNumber*: 1
- *hasCiscoInterimBuildNumber:7*
- *hasCiscoTrainIdentifier*: E

hasIdentification

product_0124

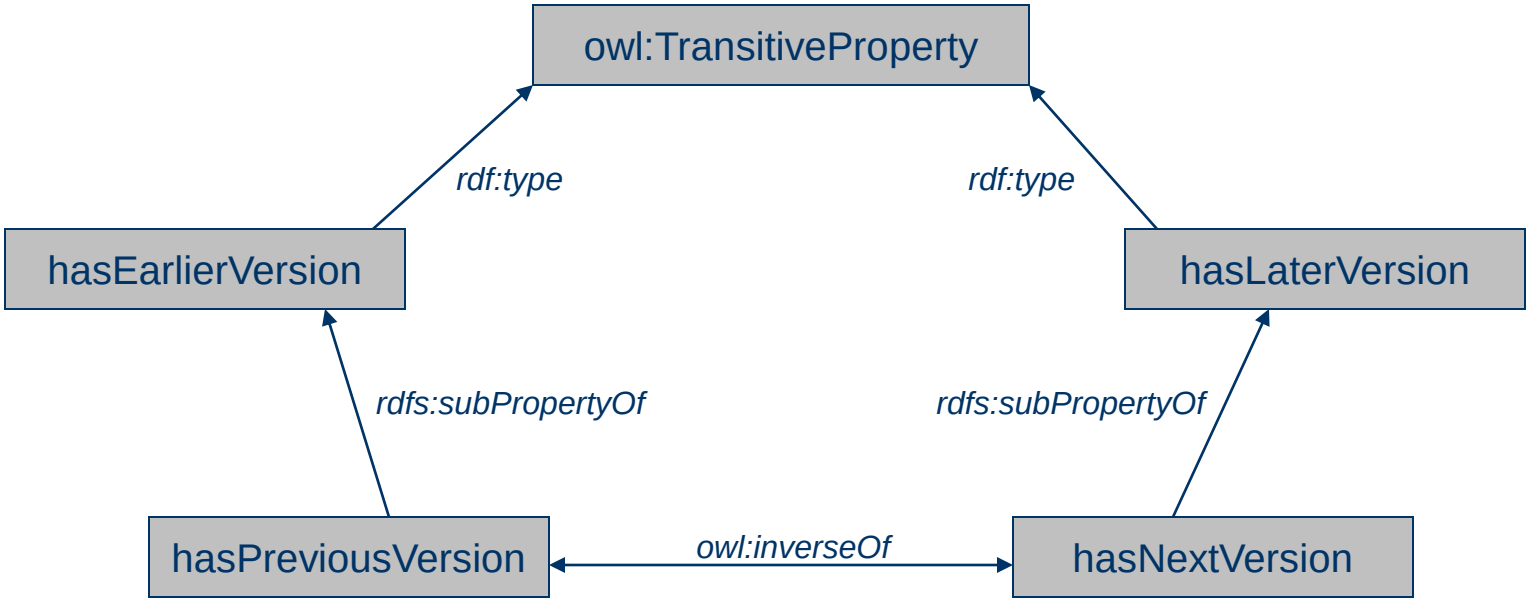
- *rdf:type*: OS
- *hasCpeName*: cpe:/o:cisco:ios:12.4(1.7)E
- *hasOwner*: Foaf_resource:Cisco

hasIdentification Property Uniquely Identifies a Product

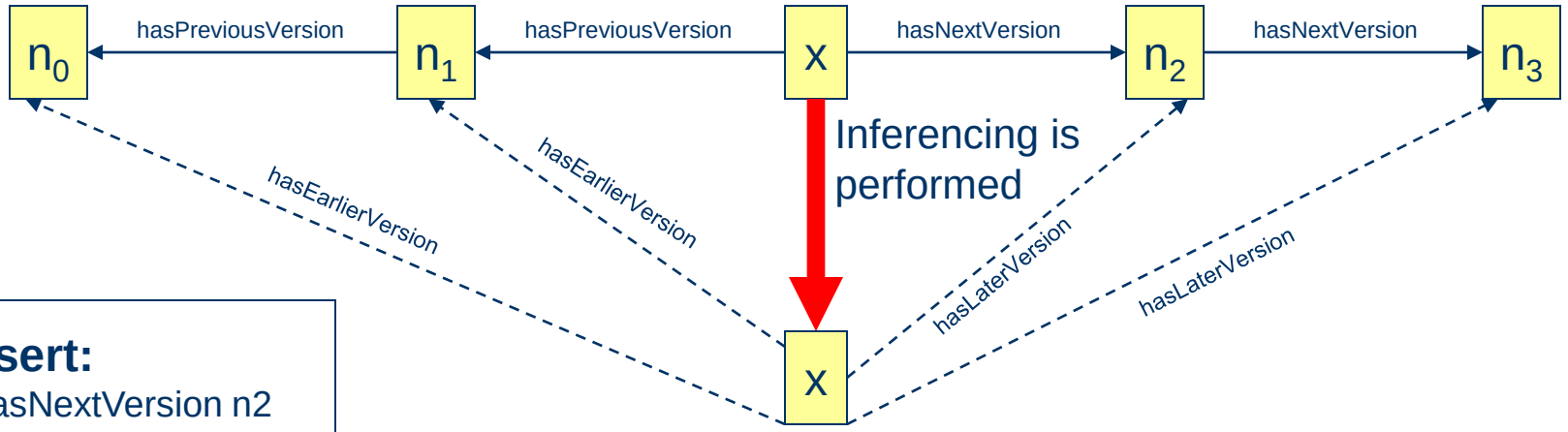


The Ontology Provides the Capability for Modeling Ranges of Products

- This is accomplished with four predicates
  - *hasNextVersion, hasPreviousVersion*
  - *hasLaterVersion (transitive), hasEarlierVersion* (transitive)
- These four predicates are modeled using a predicate hierarchy such that the non-transitive predicates are related to the transitive predicates through rdfs:subPropertyOf.



Inferencing for Product Range Data



Assert:
X hasNextVersion n2

Infer:

X hasLaterVersion n_2

n_2 hasPreviousVersion X

n_2 hasEarlierVersion X

= product instance data

= asserted triple

= inferred triple

- The reasoner creates inferred triples which allow an observer to see all products in a version chain earlier and later than x. Inferred triples are also captured for n_0, n_1, n_2, and n_3.
- The version chain DOES have to be captured by a human since a version chain order is ambiguous
  - In the future if IdentificationStrategies are modeled fully it may be possible to encode version chain order into the model and let the reasoner figure it out.

Querying for Product Range Data

- Analysts populate version chain using non-transitive predicates (hasNextVersion and hasPerviousVersion)
- A SPARQL query could then be written against the transitive predicates which the reasoner has inferred.
- Querying against the transitive predicates allow system to determine all “earlier” and all “later” versions (i.e. a product range).

```
SELECT ?product
WHERE {
        ?product a nvd:product
        ?product nvd:hasEarlierVersion 3.2
        ?product nvd:hasLaterVersion 5.4
}
```

- Keeps all application logic for range relationships in model
- This DOES require instance data to be fully populated
- Could potentially explode triples

Additional Resources

NIST websites:

- SCAP Homepage: http://scap.nist.gov
- SCAP Validated Tools: http://nvd.nist.gov/scapproducts.cfm
- National Vulnerability Database: http://nvd.nist.gov
- NIST Computer Security Resource Center (CRSC) http://csrc.nist.gov/publications/PubsSPs.html

NIST publications (available at http://csrc.nist.gov):

- Special Publication (SP) 800-126 Revision (Rev. 1), DRAFT The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.1, December 15, 2009
- SP 800-117, Draft Guide to Adopting and Using the Security Content Automation Protocol (SCAP), May 5, 2009

Contact Information

Paul Cichonski
Booz Allen Hamilton
Supporting the National Institute of Standards and Technology
paul.cichonski@nist.gov
(301) 975-8441