TOWARDS AN ONTOLOGY FOR SCA API'S

Durga Suresh and Mieczyslaw Kokar

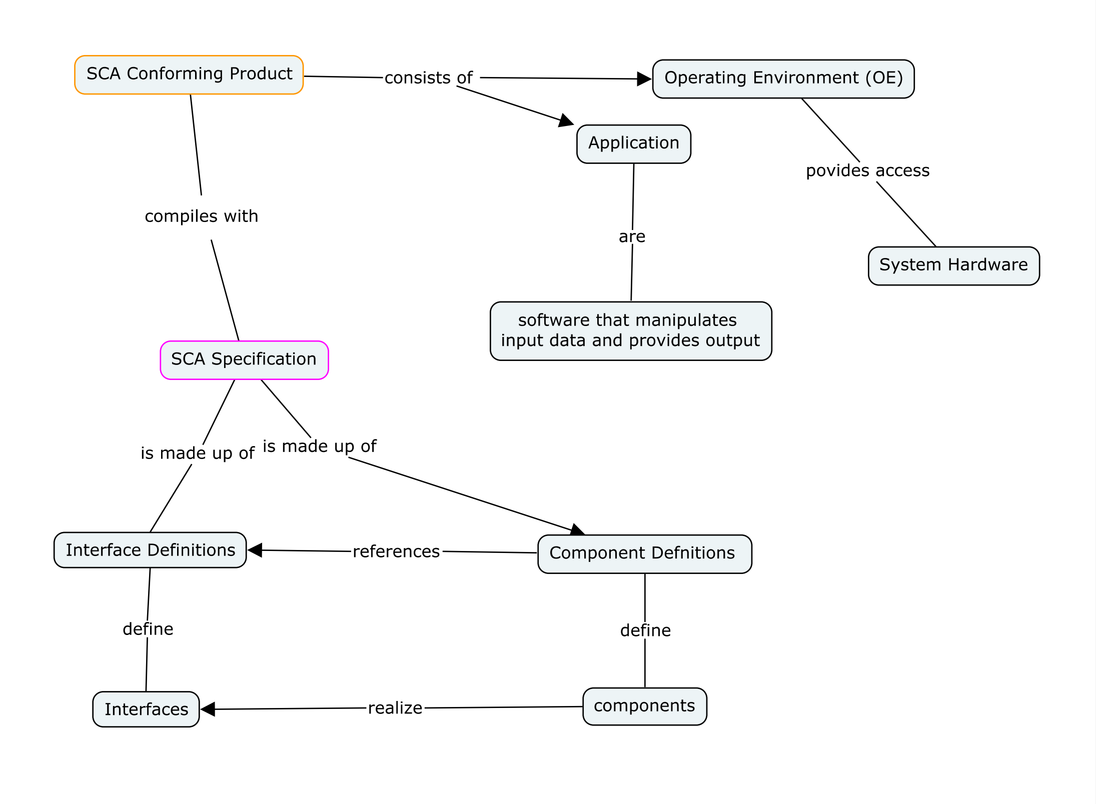
Northeastern University

AGENDA

- Introduction to SCA
- SCA Specification
- UML Vs. OWL
- The Need for an Ontology
- UML to OWL Mapping
- SCA 4.1 Specification to SCA41 Ontology
- Checking for Consistency
- Work in Progress: Querying the Specification
- Work in Progress: Writing Rules
- Conclusion
- Future Work : Prototyping
- Questions??

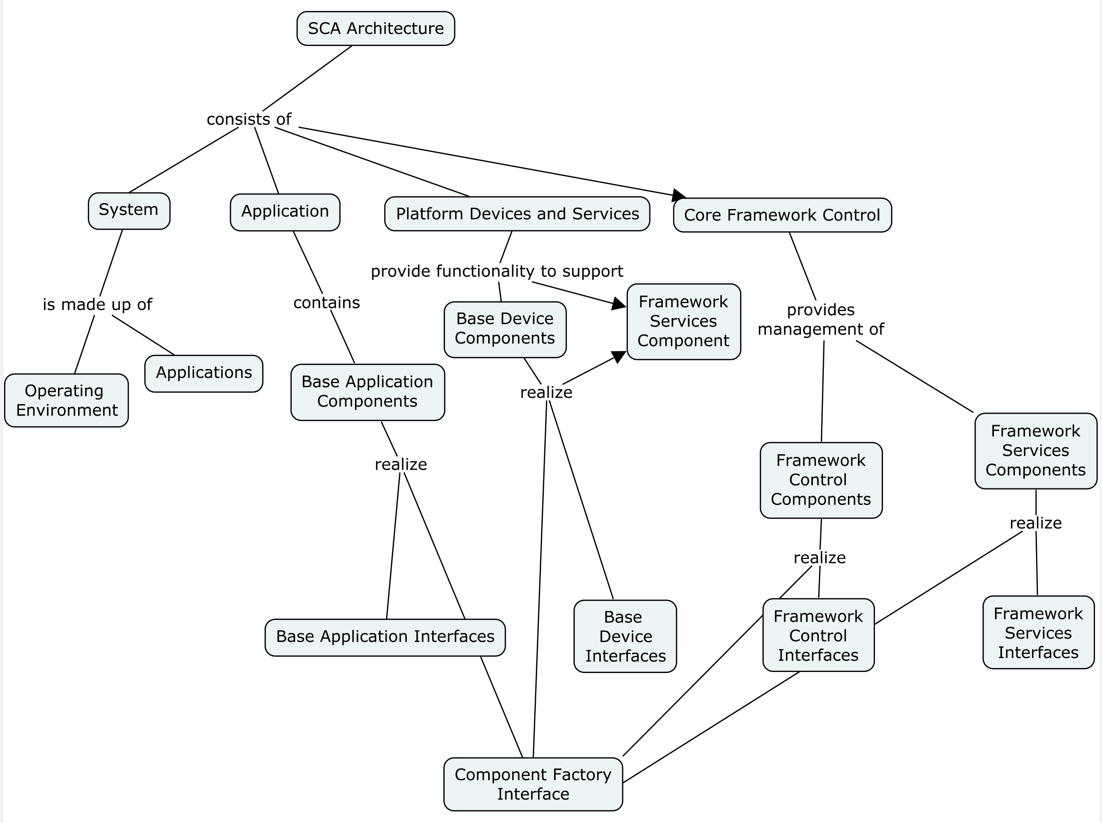
SCA 4.1 COMPONENTS AND INTERFACES

- Components definitions reference Interfaces definitions
- Components realize Interfaces
- SCA Conforming product must comply with SCA Specification
- SCA Architecture of defined using Components and Interfaces



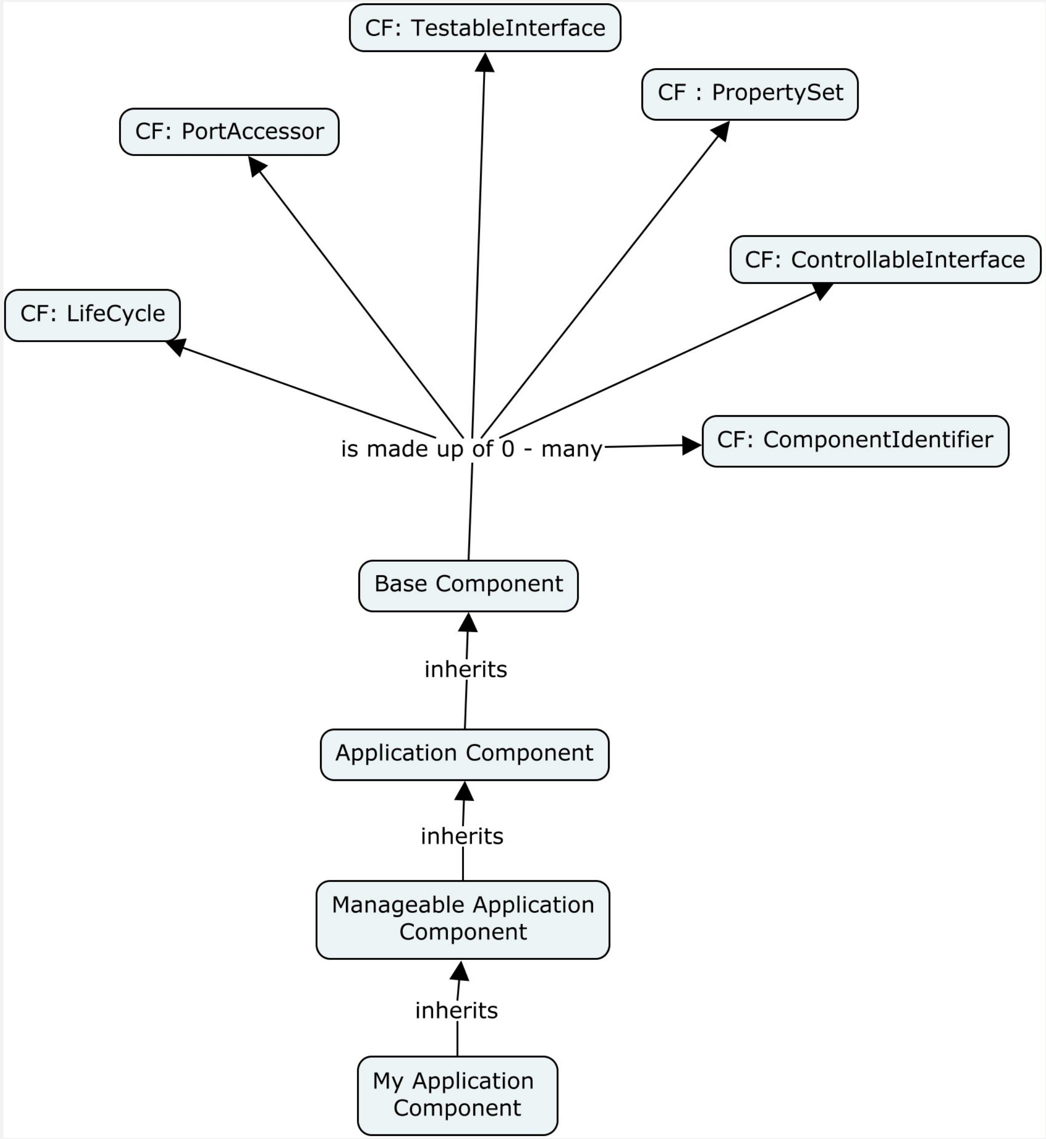
SCA 4.1 ARCHITECTURAL OVERVIEW

- SCA Architecture is composed of
  - System Architecture
  - Application Architecture
  - Platform Devices and Services Architecture
  - Core Framework Control Architecture


SCA Architecture
consists of
System
Application
Platform Devices and Services
Core Framework Control
is made up of
Operating Environment
Applications
contains
Base Application Components
provide functionality to support
Base Device Components
Framework Services Component
provides management of
realize
realize
Framework Control Components
Framework Services Components
realize
realize
Base Application Interfaces
Base Device Interfaces
Framework Control Interfaces
Framework Services Interfaces
Component Factory Interface

SCA 4.1 INHERITANCE STRUCTURE OF COMPONENTS

- Another view of the SCA 4.1 Specification showing the structure of creating an Application Component
- SCA is a component-based architecture where component refers to a piece of software



NEED FOR AN ONTOLOGY

- No open-source implementation of the SCA since OSSIE. Many COTS solutions are available.
- There is continuous development of API's for the SCA
- It is difficult to automatically analyze the Specification
- UML does not provide inference capabilities
- Prototyping the API without implementation is not possible

UML TO OWL MAPPING

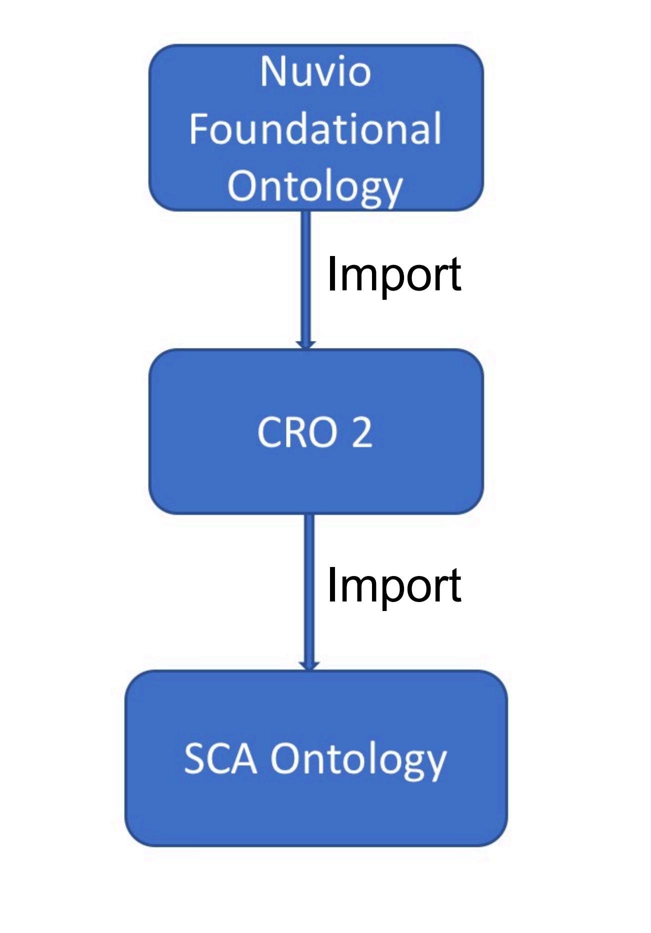
UML Elements	OWL Elements
Class, property owned attribute, type	Class
Instance	Individual
ownedAttribute, binary association	Property
Subclass	subClassOf
Generalization	subProperty
N-ary association, association class	Class, Property
Enumeration	oneOf
Disjoint, cover	disjointWith, unionOf
Multiplicity	minCardinality maxCardinality
Package	Ontology
Dependency	Reserved name rdf:Property

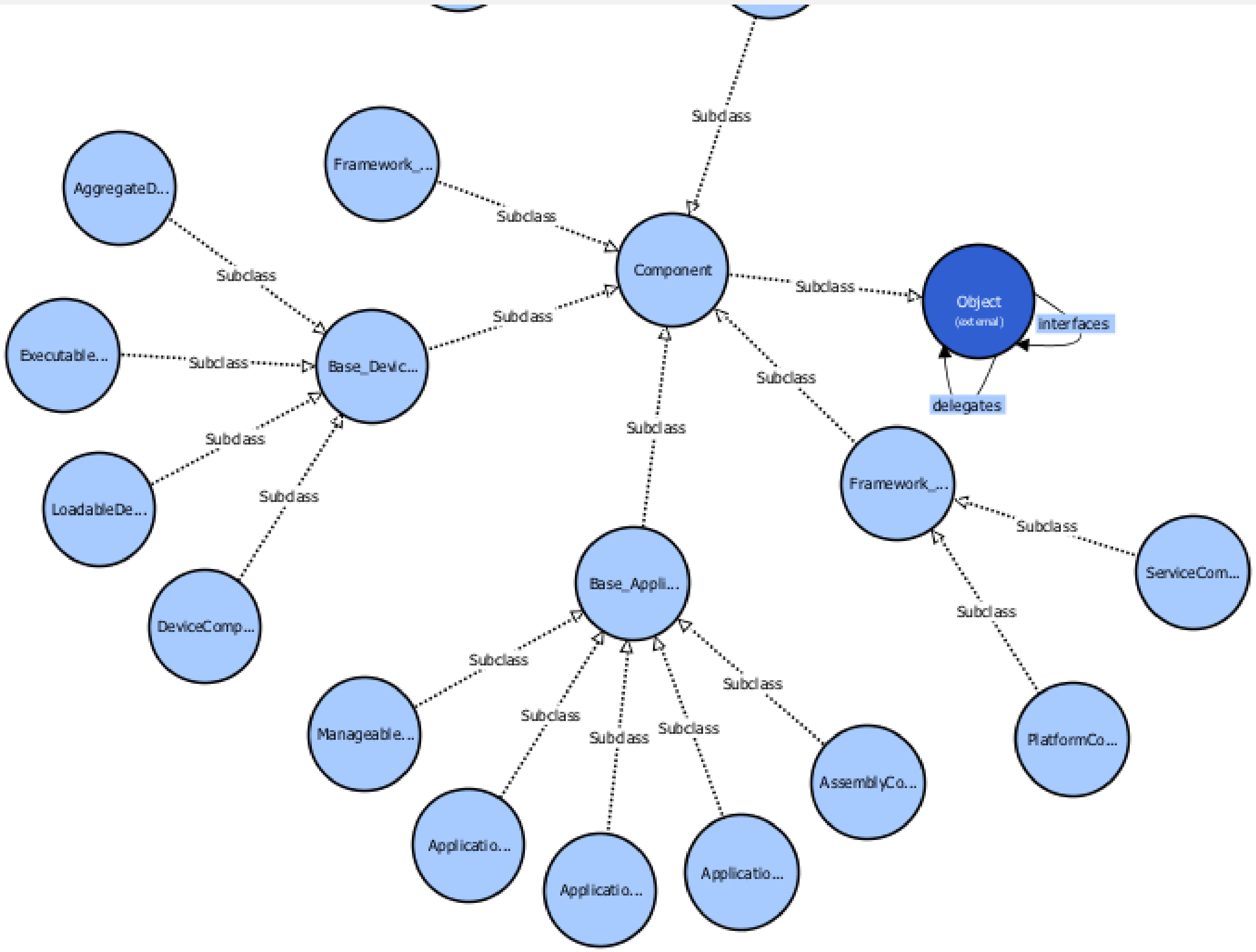
SCA 4.1 SPECIFICATION TO SCA41 ONTOLOGY

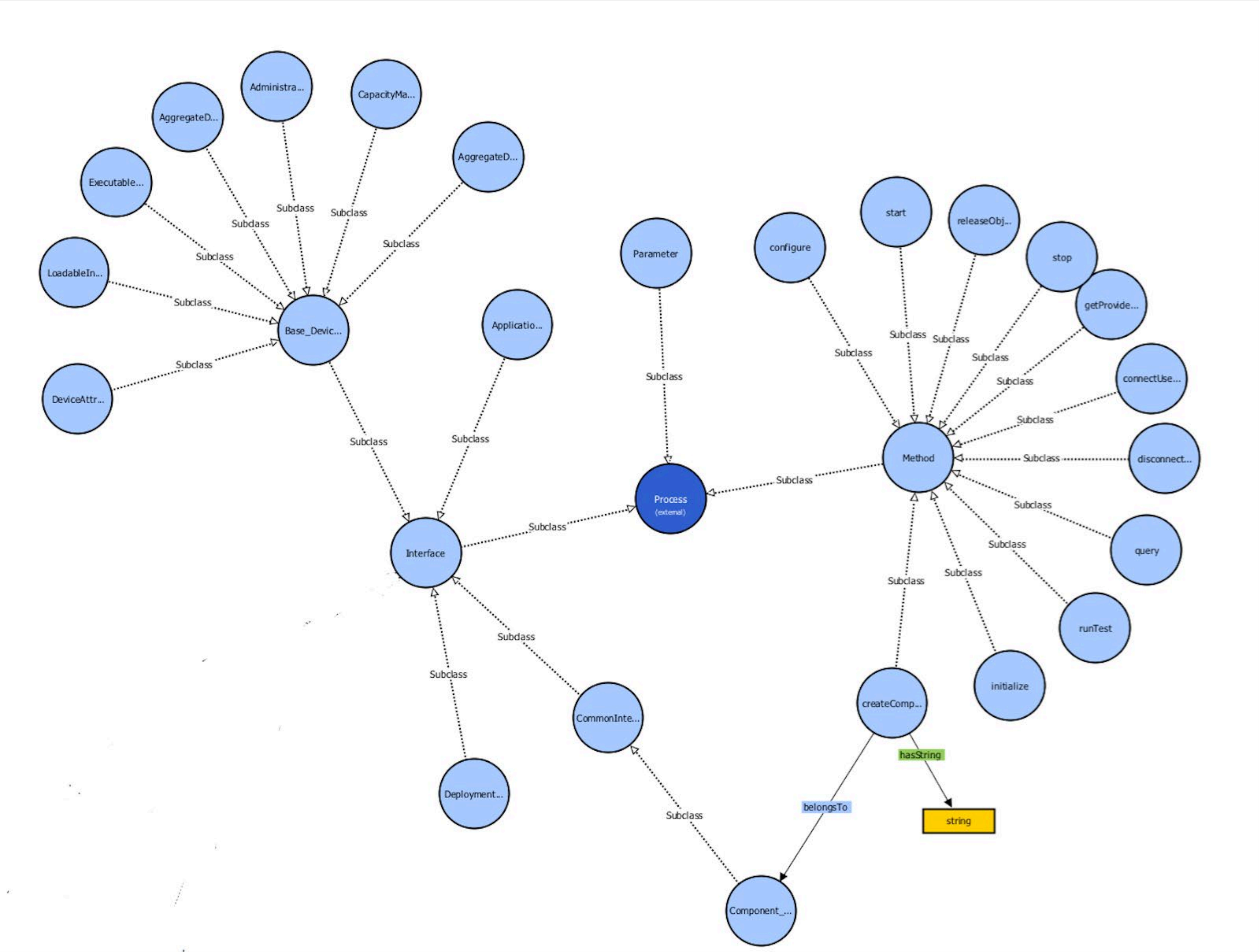
Association	Object Property
Interrogable	owl:topObjectProperty-> hasParticipant-> interrogable
Testable	owl:topObjectProperty-> hasParticipant-> testable
Controllable	owl:topObjectProperty-> hasParticipant-> controllable
Releasable	owl:topObjectProperty-> hasParticipant-> releaseable
Configurable	owl:topObjectProperty-> hasParticipant-> configurable
Connectable	owl:topObjectProperty-> hasParticipant-> connectable
creates	owl:topObjectProperty-> creates
delegates	owl:topObjectProperty-> delegates
interfaces	owl:topObjectProperty-> interfaces
accesses	owl:topObjectProperty-> accesses

SCA 4.1 SPECIFICATION TO SCA4I ONTOLOGY

- Nuvio Foundational Ontology is imported into CRO2
- CRO2 is imported into SCA Ontology
- CRO2 defines the terms in the Cognitive Radio Domain




Subclass
Framework_...
AggregateD...
Subclass
Subclass
Component
Subclass
Object
(external)
interfaces
Executable...
Subclass
Base_Devic...
Subclass
delegates
Subclass
Subclass
Subclass
LoadableDe...
Framework_...
Subclass
Subclass
ServiceCom...
DeviceComp...
Base_Appli...
Subclass
Subclass
Subclass
Manageable...
Subclass
Subclass
Subclass
PlatformCo...
AssemblyCo...
Applicatio...
Applicatio...
Applicatio...


Administra...
AggregateD...
CapacityMa...
AggregateD...
Executable...
LoadableIn...
DeviceAttr...
Base_Devic...
Applicatio...
Interface
Deployment...
CommonInte...
Component_...
Parameter
Process
(external)
configure
start
releaseObj...
stop
getProvide...
connectUse...
disconnect...
query
runTest
initialize
createComp...
Method
hasString
belongsTo
string
Subclass

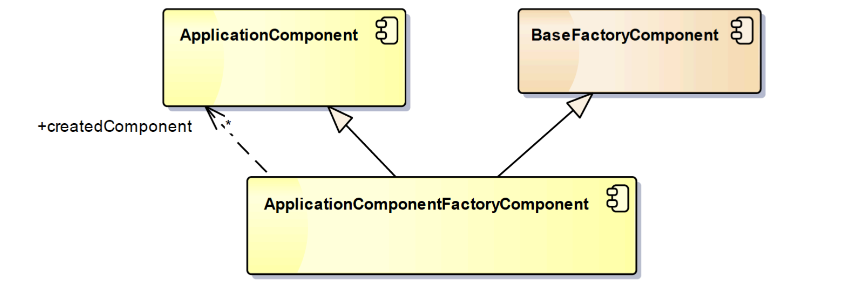
UML VS. OWL METRICS

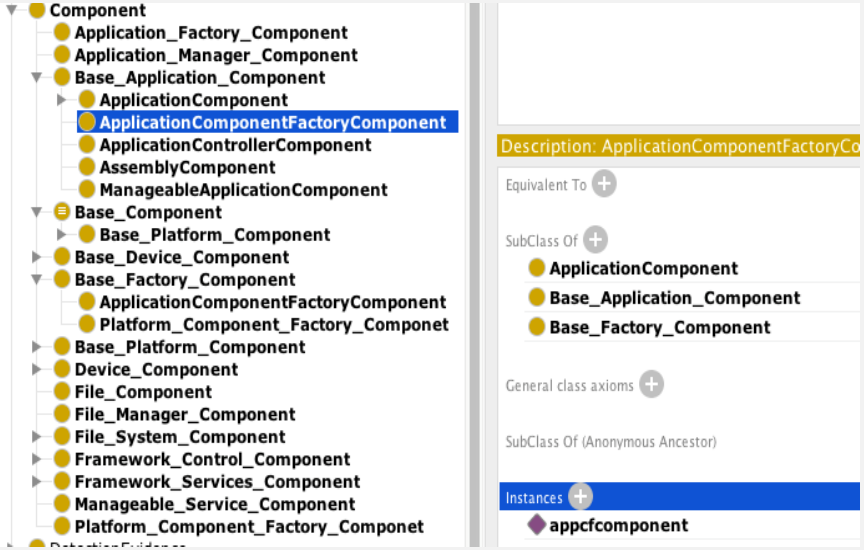
- 26 UML Class diagrams that describe Interfaces
- 22 UML Class diagram that model component

Axioms	1149
Logical axiom count	643
Declaration axiom count	428
Class count	278
Object property count	111
Data Property count	20
Individual count	20

CHECKING FOR CONSISTENCY

- UML Specification is checked for consistency
- Class diagram shows the inheritance structure of Application Component Factory Component
- Ontology shows the structure of mapping
- Reasoner will show inconsistency when inheritance relation is not satisfied




Component
Application_Factory_Component
Application_Manager_Component
Base_Application_Component
ApplicationComponent
ApplicationComponentFactoryComponent
ApplicationControllerComponent
AssemblyComponent
ManageableApplicationComponent
Base_Component
Base_Platform_Component
Base_Device_Component
Base_Factory_Component
ApplicationComponentFactoryComponent
Platform_Component_Factory_Componet
Base_Platform_Component
Device_Component
File_Component
File_Manager_Component
File_System_Component
Framework_Control_Component
Framework_Services_Component
Manageable_Service_Component
Platform_Component_Factory_Componet

Description: ApplicationComponentFactoryCo

Equivalent To

SubClass Of
ApplicationComponent
Base_Application_Component
Base_Factory_Component

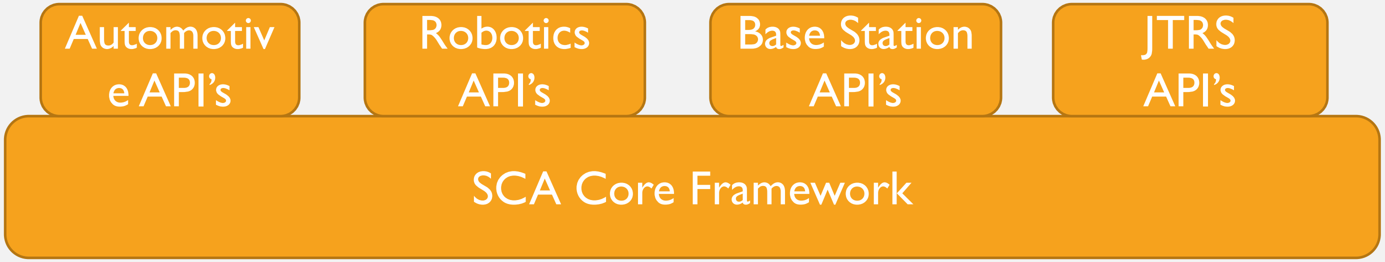
General class axioms

SubClass Of (Anonymous Ancestor)

Instances
appcfcomponent

SCA API'S

- SCA is independent of the application domain
- Different applications are supported by different domain-specific API's



SCA API'S

- Partial list of JTRS API's

Audio Port Device API	Ethernet Device API
Frequency Reference Device API	GPS Device API
Modern Hardware Abstraction Layer (MHAL) API	Serial Port Device API
Timing Service API	Vocoder Service API
MHAL On Chip Bus (MOCB) API	Packet API
JTRS Platform Adapter (JPA) API	

WIP: QUERYING THE SPECIFICATION

WIP: RULES AND CONSTRAINTS

- Map one of the API's of the JTNC e.g. the Transceiver API to SCA41 Ontology (we have already done this). Write Queries to check for correctness of the relationships and dependencies
- Write rules that can show how the API works
- Rules and Constraints will be written in BVR

LESSONS LEARNED

- SCA41 Ontology enables:
  - Checking for consistency and querying
  - Can establish a standard with the radio domain
  - No need to implement API’s individually

FUTURE WORK

- Prototype the API's in the SCA
- Writing use cases that include SDR domain and SDN domain will help clarify the need for the Ontological approach

FUTURE OF SCA

- Future of SCA is unclear from yesterdays panel.
- The idea of having multiple framework was discussed and seemed to be the future
- Prototyping will still be useful and hence should be explored

THANK YOU FOR YOUR TIME!
QUESTIONS???