

# Governing Delegation of Authority within SOA Environments Using KAoS



**ROBERT L. SEDLMEYER, IPFW**  
**JIM JACOBS, RAYTHEON**  
**ANDRZEJ USZOK, IHMC**  
**JAMES MILLIGAN, AFRL**

# Objectives

2

- **Design, develop, and demonstrate a delegation of authority access control service for Service Oriented Architectures (SOA)**
  - Capture delegation policies in a semantic model from which delegation policies can be specified
  - Develop delegation policy enforcement mechanisms
  - Maximize interoperability
  - Minimize impact on existing services
  - Demonstrate capability in an operationally relevant scenario

# Operational Context

3

Human-in-the-loop Service Orchestration



User 123

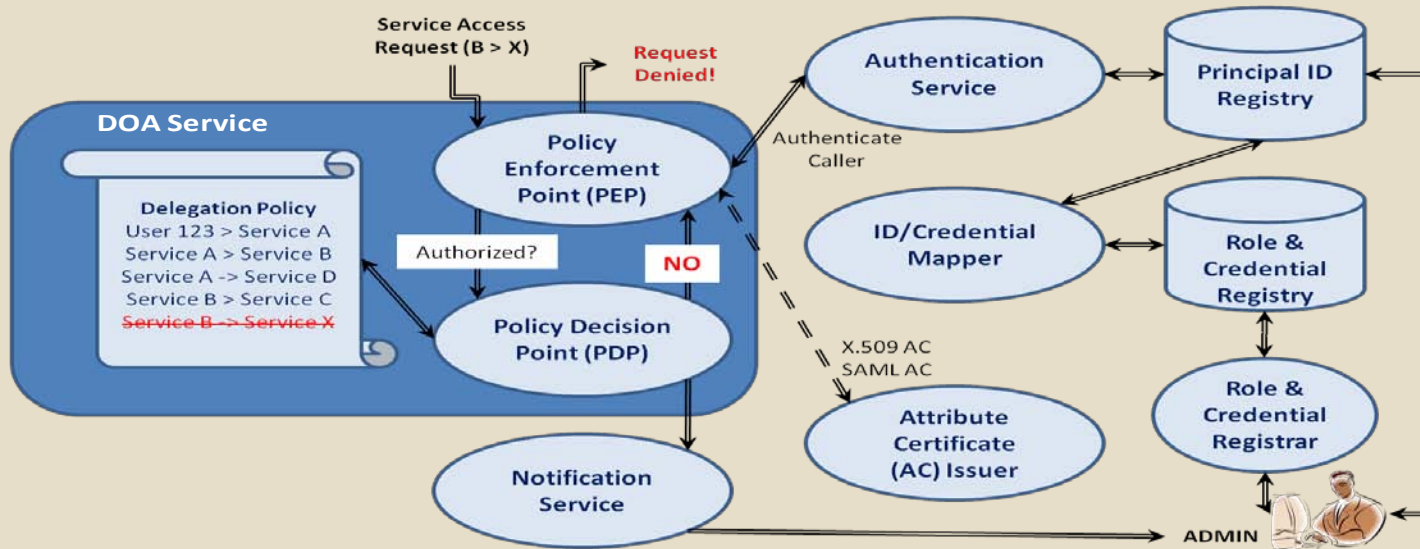
Service A

Service B

Service C

Service X

Service D



# Technical Approach

4

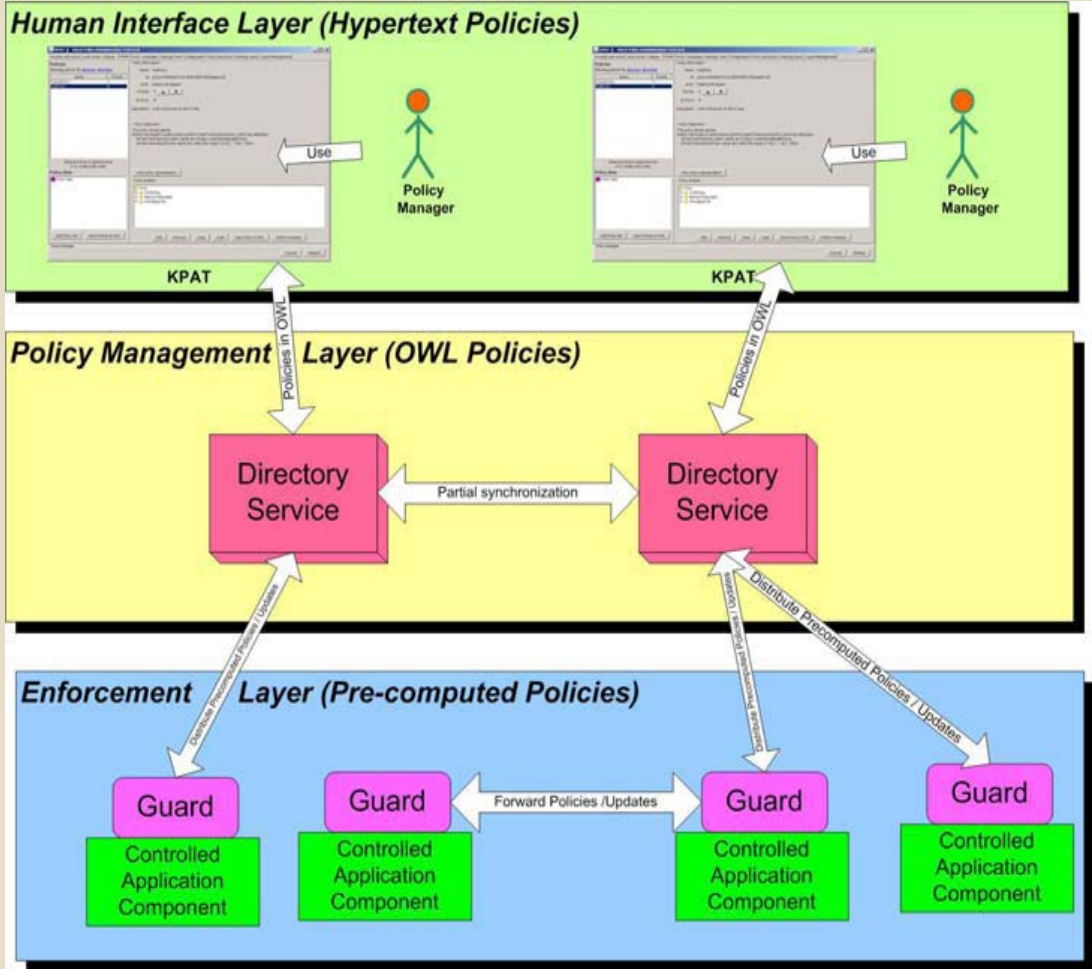
- Assumptions
  - ✦ Authentication is handled separately from authorization
    - ...but authentication info carried in X.509 certificate
  - ✦ Delegation of authority is role-based, but could have contextual constraints
- Capture delegation policies in a semantic model from which delegation policies can be specified
  - ✦ Define ontologies that model operational domain, access control policies and delegation of authority policies
  - ✦ Utilize KAoS policy language
- Develop delegation policy enforcement mechanisms
  - ✦ Utilize KAoS policy engine

# Technical Approach

5

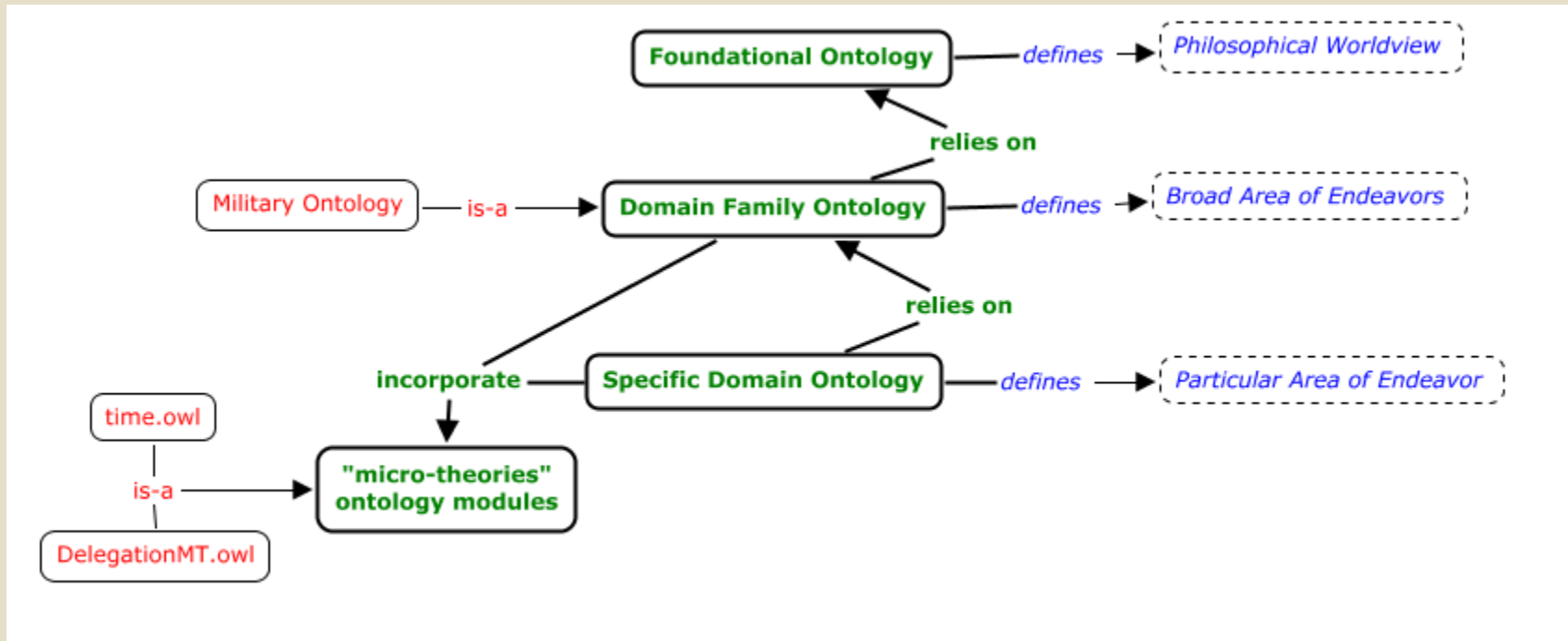
- Maximize interoperability
  - ✦ OWL for semantic modeling
  - ✦ WSDL for web service description
- Minimize impact on existing services
  - ✦ Where possible, restrict access control specifications to web service interface components
  - ✦ Place run-time code for policy enforcement in handlers

# About KAoS



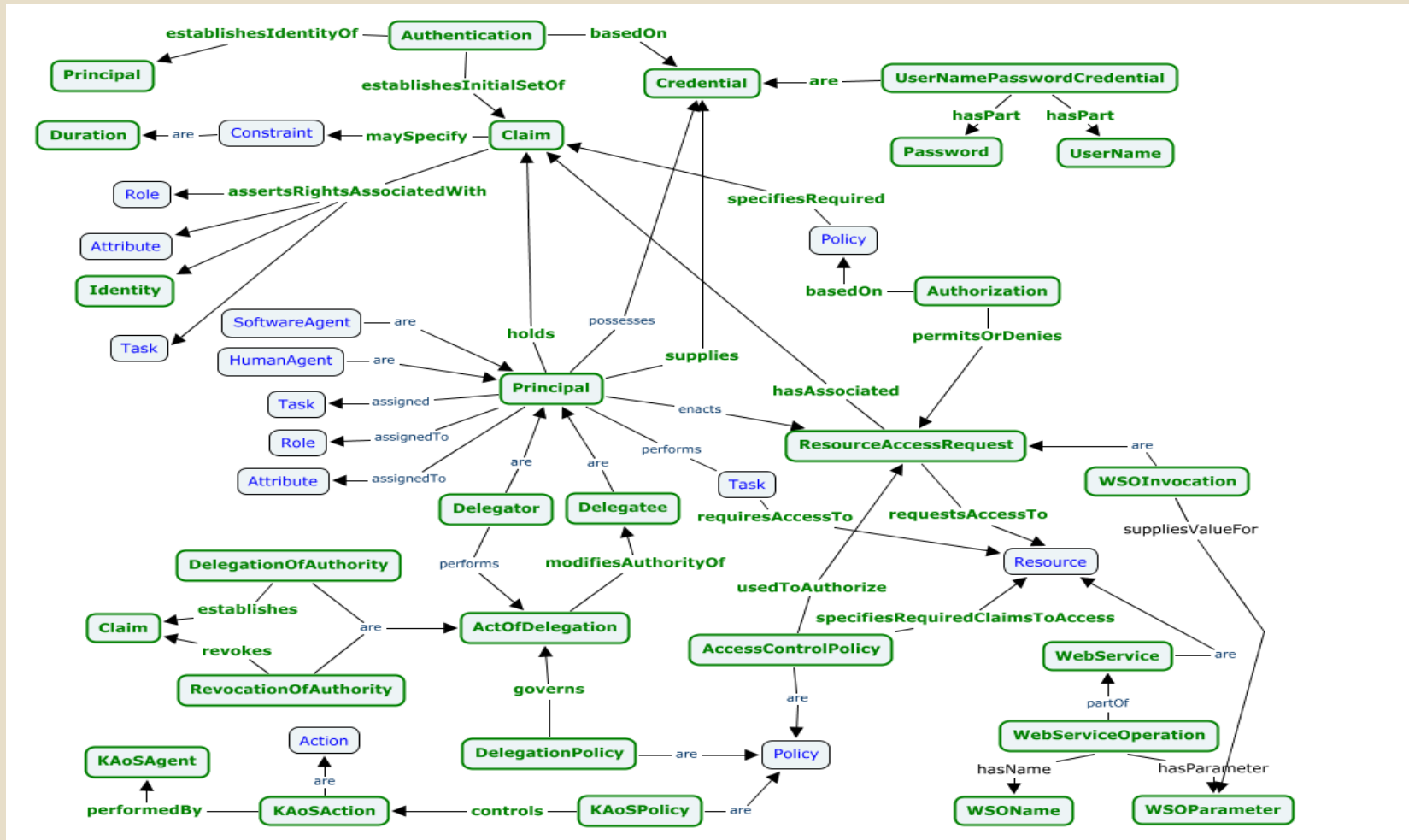
# Semantic Model: Using a Foundational Ontology

7



# Semantic Model: Delegation Entity Relations

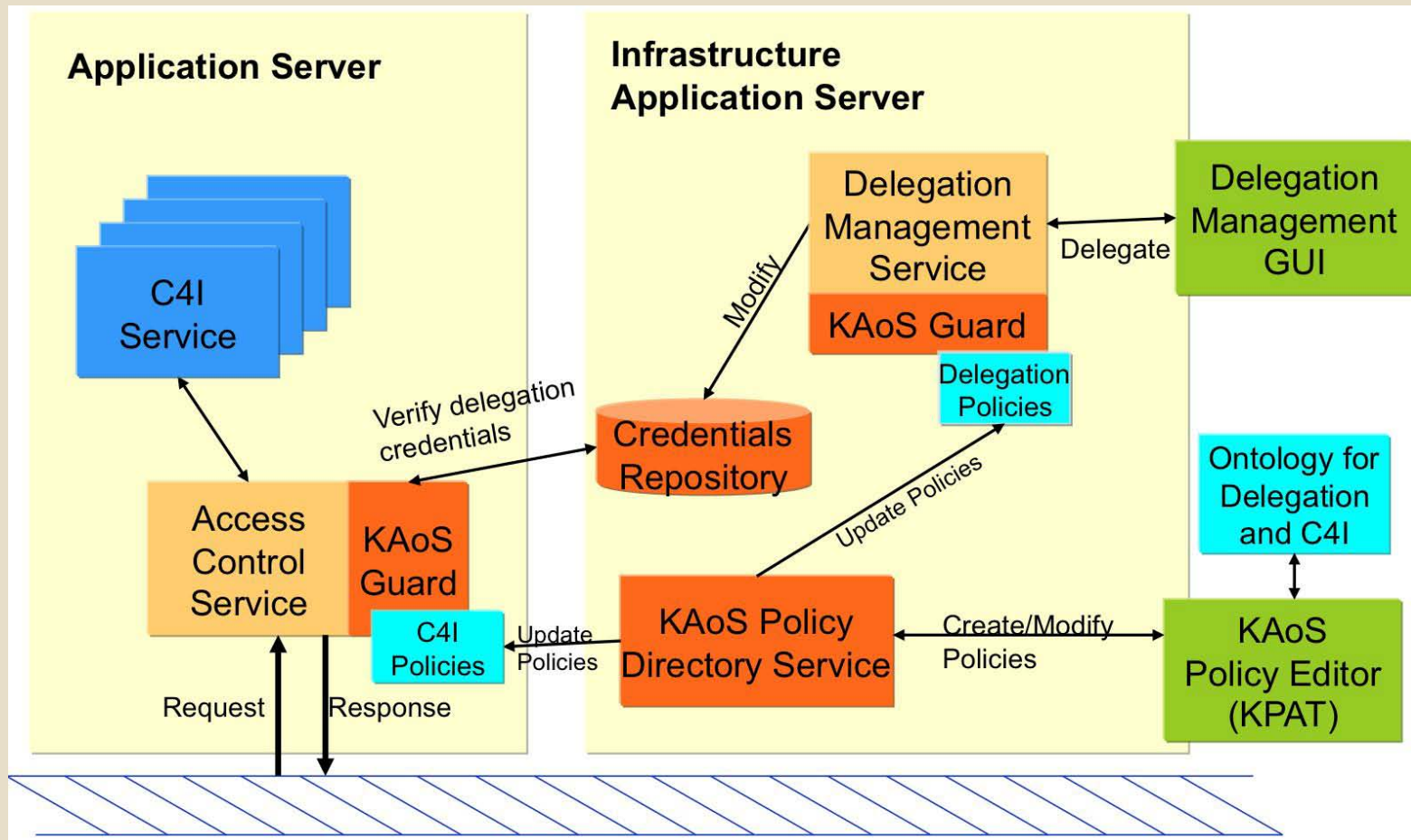
8





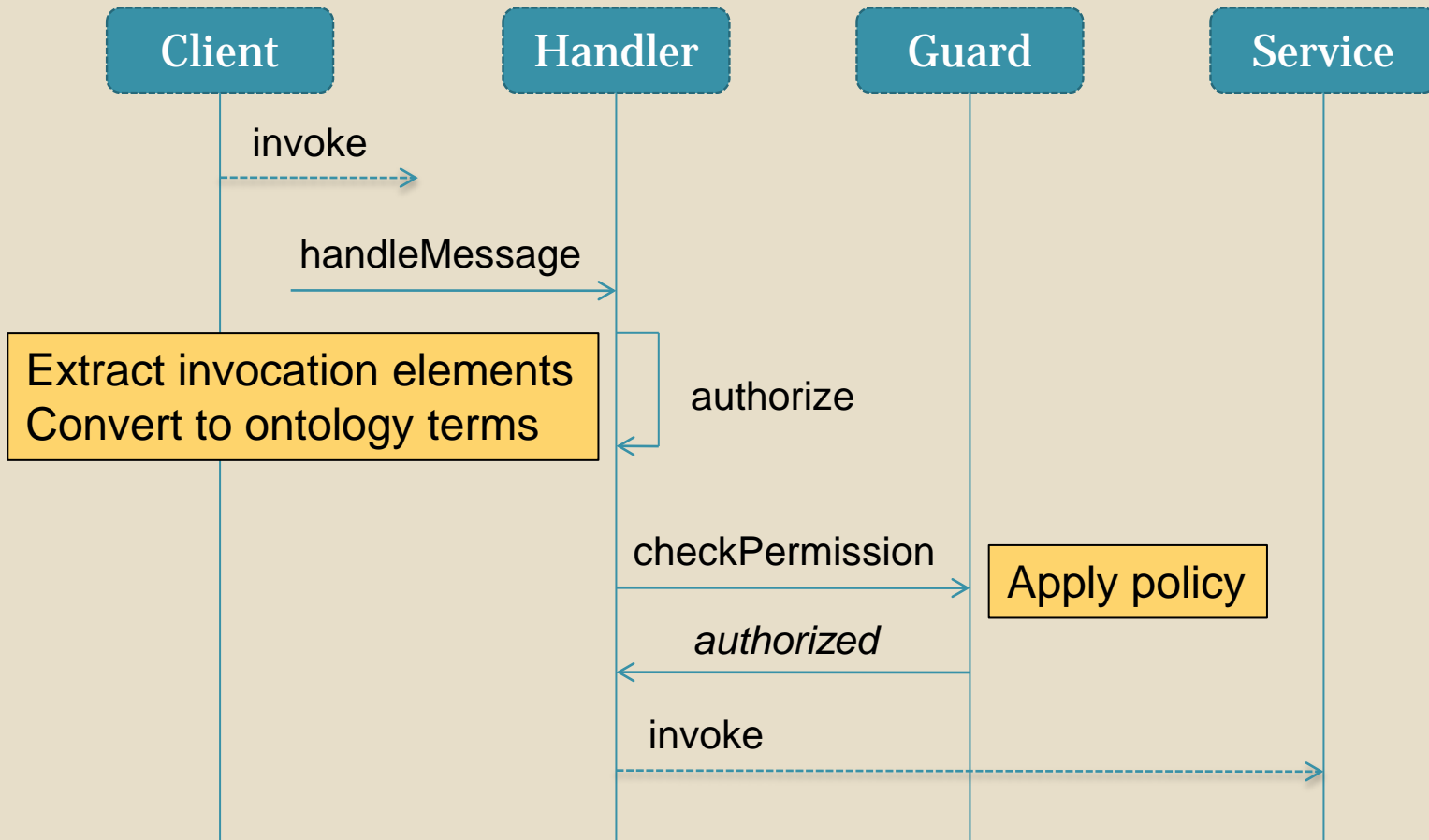
# Demonstration Architecture

9



# Authorization Control Flow

10



# Example

11

601st Air Operations Center  
Combat Operations Division

Welcome smith. [Logout](#)

Delegation Management Console

Current Delegations

User	Delegated Role	Action
baker	TargeteerRole	<a href="#">Revoke</a>

Select User  Select Role

Status: <http://ontology.ihmc.us/Military/MilitaryEntity.owl#TargeteerRole> delegated to urn:KAoS#baker

- The Delegation web service implements role delegation operations
- Objective is to ensure that only personnel serving in ‘Senior...Officer’ roles have access its operations
- KAoS policy created using KPAT

*“Any SeniorIntelligenceDutyOfficer is permitted to delegate the Targeteer role to any IntelligenceOfficer.”*

# Future Work

12

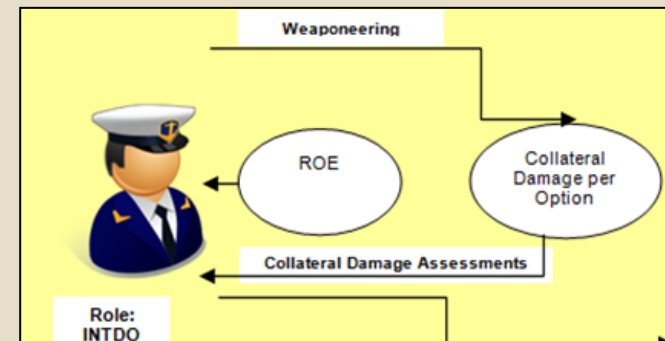
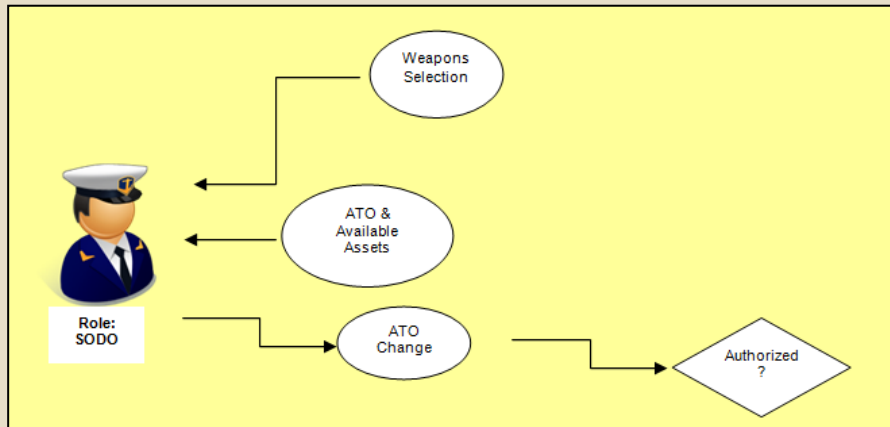
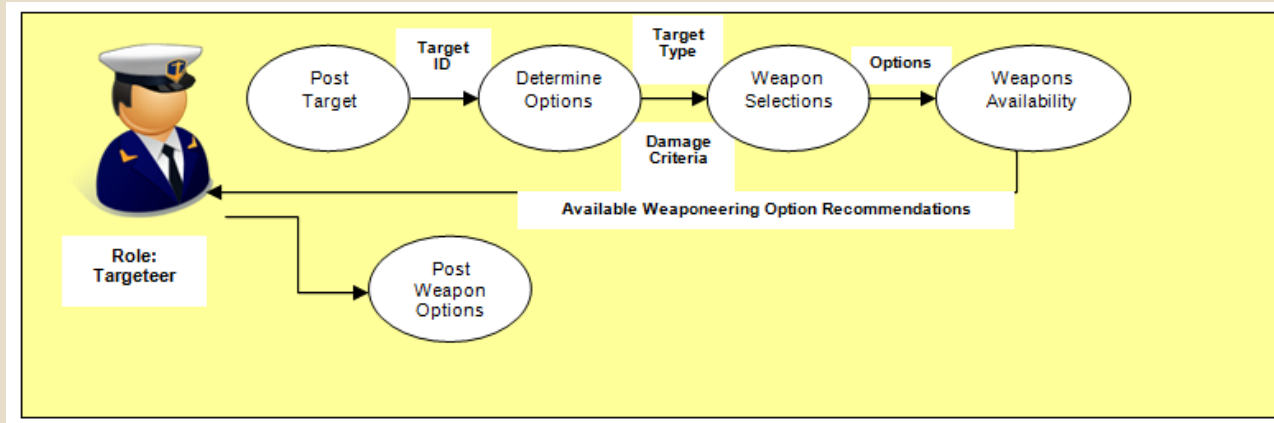
- Develop tool for automatically adding SAWSDL annotations (prototype exists)
- Extend and refine micro-theory of delegation-of-authority
- Enhance usability of KPAT to aid in construction of more sophisticated policies
- Make architecture more flexible so other authentication mechanisms are easily integrated

# Backup Slides

13

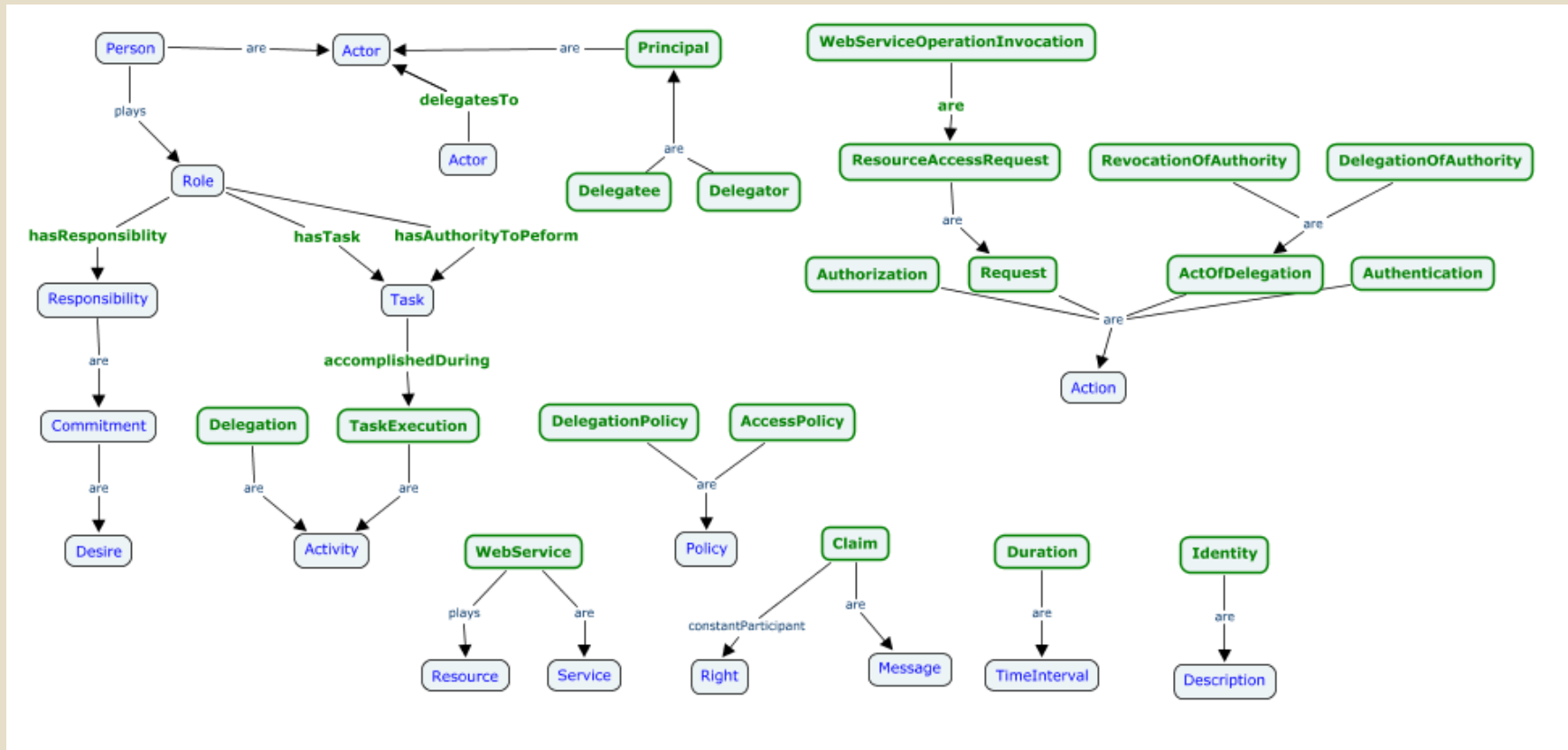
# Demonstration Scenario

14



# Semantic Model: Delegation Entity Types

15



# DelegateRole Implementation Details

16

```
// convert KAoS MilitaryRole term represented by delegated role parameter to
// corresponding KAoS MilitaryActor term
delegatedRole = delegatedRole.substring(MilitaryEntityConcepts.MilitaryEntityOwlURL().length());
delegatedRole = MilitaryActorConcepts.MilitaryActorOwlURL() + delegatedRole;
delegatedRole = delegatedRole.substring(0, delegatedRole.indexOf("Role"));
// obtain handle to KAoS Actor associated with delegatee
QueryRegistration myQueryRegistration = CSIFactory.getQueryRegistration();
KAoSAgentDescription myDelegateeActor = myQueryRegistration.getActor(delegateeId);
// add new Actor type to delegatee
myDelegateeActor.addEntityOntologicalType(delegatedRole, "");
// record delegated action
myDelegationActionID = (String) mySOAPMessageContext.get("actionId");
ActionInstanceDescription action = (ActionInstanceDescription)mySOAPMessageContext.get("action");
delegationRepository.put(myDelegationActionID, action);
```

**Unlike domain web services, the Delegation service modifies ontology instance data maintained by KAoS**



# WSDL Modifications

17

```
<xsd:element name="DelegateRole"
  sawsdl:modelReference="http://ontology.ihmc.us/DelegationAction.owl#DelegationAction"
  sawsdl:liftingSchemaMapping="DelegateRole2Ont.xsl">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="delegateeId" type="xsd:string" />
      <xsd:element name="delegatedRole" type="xsd:string" />
      <xsd:element name="delegationContext" type="xsd:string" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

- SAWSDL (Semantic Annotations for WSDL and XML Schema) is a W3C Recommendation for mapping WSDL components to ontology models.
- Here we use it to define an XSL file that contains the translation rules for mapping the DelegateRole service operation to our policy ontology.

# Delegation Service Annotation

18

```
@WebService(  
    endpointInterface = "com.ray.ont.delegation.DelegationService",  
    targetNamespace = "http://ont.ray.com/Delegation/",  
    serviceName = "DelegationService", portName = "DelegationServiceSOAP",  
    wsdlLocation = "/WEB-INF/wsdl/delegation.wsdl"  
)
```

```
@HandlerChain(file = "/resources/handlerchain.xml")  
public class DelegationServiceImpl implements DelegationService  
{  
:  
}
```

**@HandlerChain used to associate web service with JAX-WS Handler that contains KAoS Guard for performing policy checking**

# DelegateRole2Ont.xsl

19

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:action="http://ontology.ihmc.us/DelegationAction.owl#"
  xmlns:ns2="http://ont.ray.com/Delegation/"
  xmlns:java="http://xml.apache.org/xalan/java"
  exclude-result-prefixes="java">
  <xsl:template match="ns2:DelegateRole">
    <rdf:Description rdf:about="REPLACE-WITH-KAOS-URI">
      <rdf:type rdf:resource="http://ontology.ihmc.us/DelegationAction.owl#DelegationAction"/>
      <action:hasDelegatedRole rdf:resource="{delegatedRole}"/>
      <action:hasDelegee rdf:resource="{delegateeId}"/>
      <action:hasDelegationContext rdf:resource="{delegationContext}"/>
    </rdf:Description>
  </xsl:template>
</xsl:stylesheet>
```

Mapping rules specify how service operation parameters are mapped to ontological terms