Japan-Austria Joint Workshop on “ICT”

October 18-19 2010, Tokyo, Japan

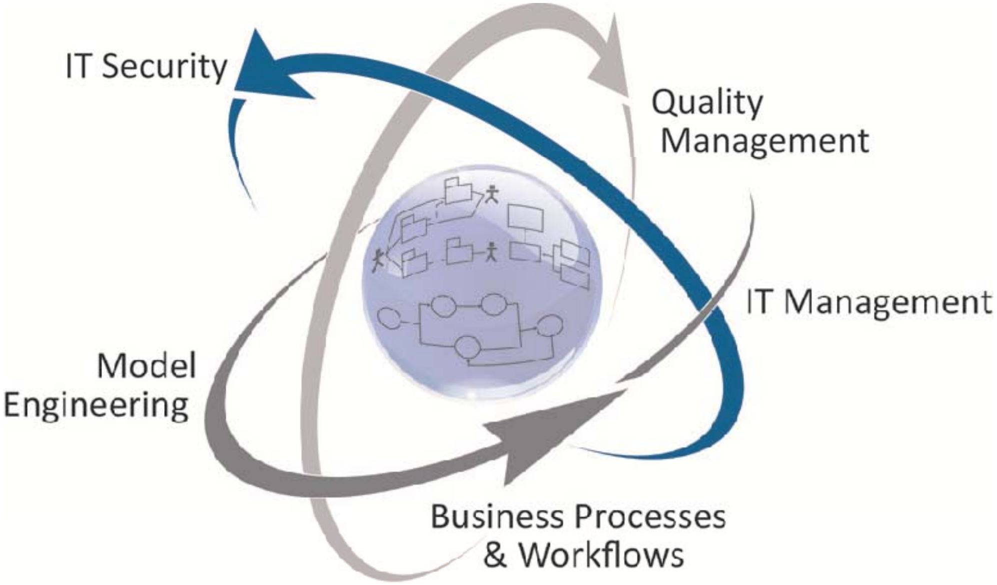
SECTET-
Model driven Security of Service Oriented Systems based on Security-as-a-Service

Basel Katt, Ruth Breu, Mukhtiar Memon and Michael Hafner

Research Group Quality Engineering

University of Innsbruck

Quality Engineering



Selected Projects

QE LaB

secure CHANGE

SoftNet

PoSecCo

COSEMA
Collaborative Security Management

tts
telling test stories

QE LaB

Quality Engineering Laura Bassi Lab

Living Models for Collaborative Systems

CENTRES OF EXPERTISE LAURA BASSI

Industry Partners

phion

ITH icoserve technology for healthcare a siemens company

kapsch >>> always one step ahead

iteratec

QE QUALITY ENGINEERING

Agenda

- Motivation
  - Service Oriented Systems
  - Challenges
- Healthcare Scenario
- SECTET : Model based configuration of Service Oriented Systems
  - Model Driven Security (MDS)
  - Security as a Service (SeAAS) Architecture
- Conclusion

Service Oriented Systems

- Independent partners offer and call services
- Collaboration across enterprises and systems
- New generation of cooperative applications
  - Electronic health record, traffic management, energy trading, etc.

Challenges

- Collaborative systems based on SOA
  - Dynamically composed, language and technology independent
  - Agile and dynamically evolving systems
- Standards only address basic security requirements
  - Solve these requirements at a low technical level
- Security enforcement at the service end points
  - Places significant processing burden on service nodes
  - Renders maintenance and management cumbersome

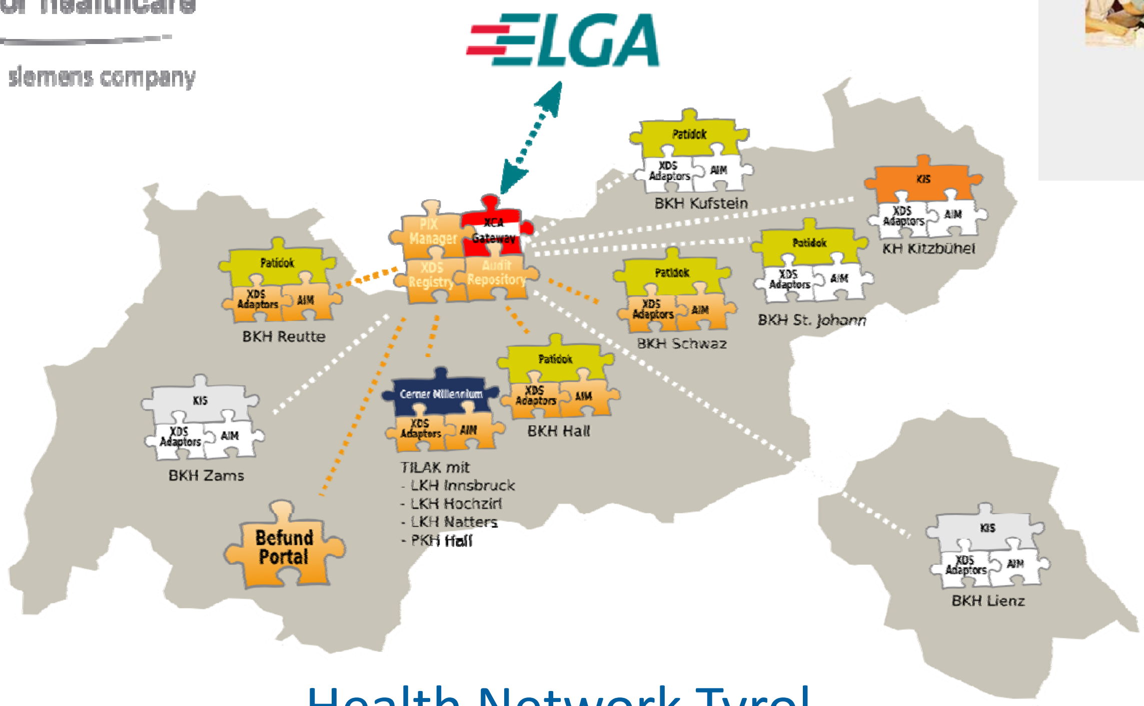
Goals

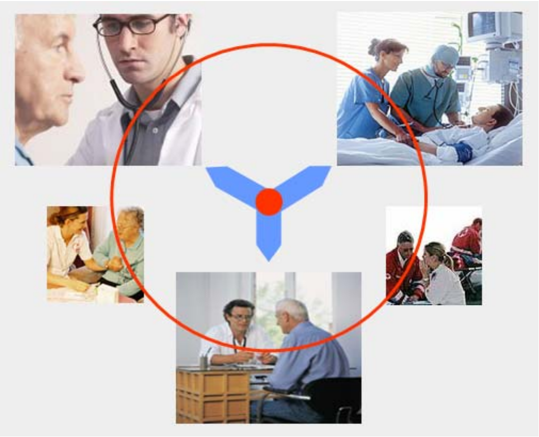
- The gap between domain experts and software engineers
- Maintainability and configurabl„ity“ of security services
  - Ability to re-configure after deployment due to requirement changes or mechanisms‘ updates
  - Support of multiple security architectures for each requirement
- Enforcement
  - Enforcing complex security requirements
  - Consistent enforcement of security policies in enterprise-level solutions
- Performance
  - Security services involve performance costly functions

Example – Distributed Electronic Health Record (EHR)

sense
smart ehealth solutions

ITH icoserve
technology for healthcare
a siemens company

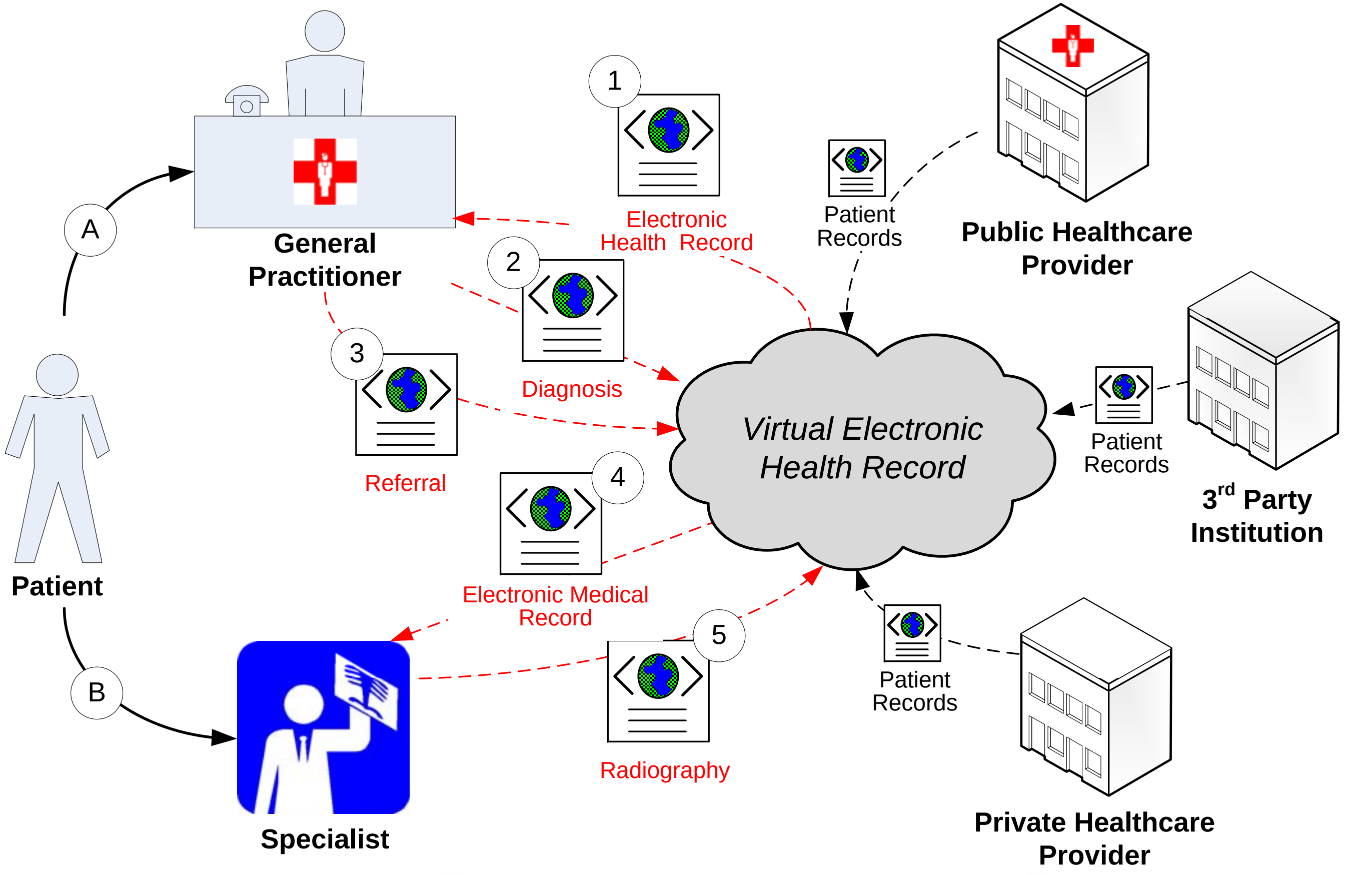




Health Network Tyrol

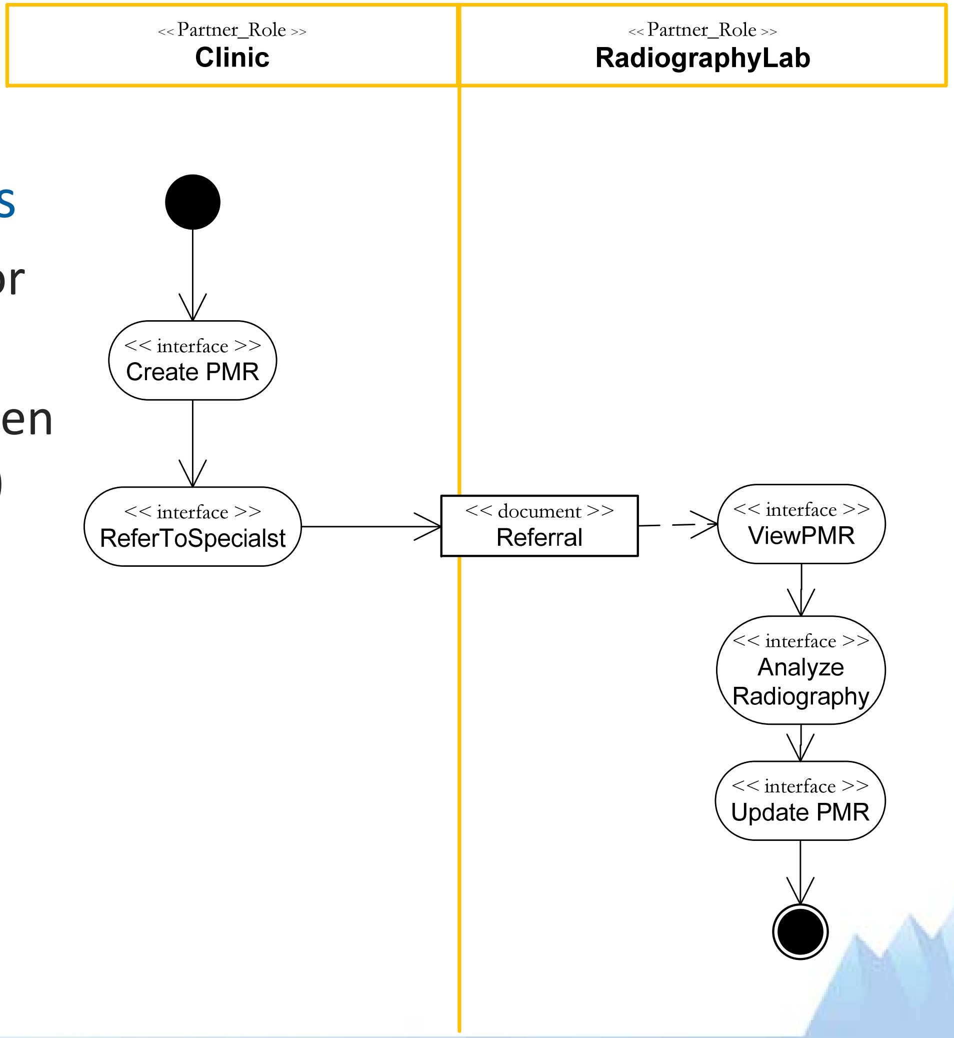
Example – Healthcare Scenario

- EHR represents a consolidated virtual medical record
  - Distributed across various care providers

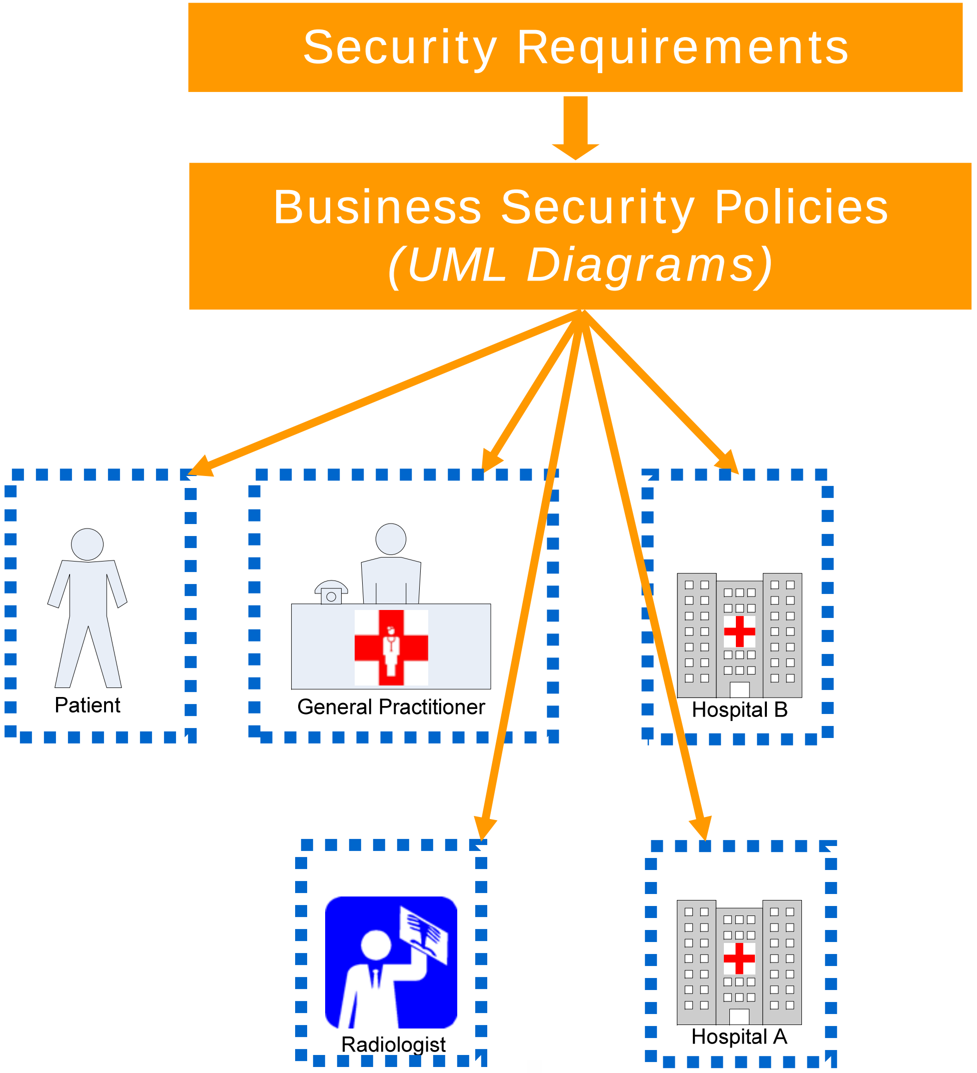


Example – Healthcare Scenario

- Inter-organizational workflows
  - Services that can be offered or called by each partner
  - Functional interaction between different stakeholders (roles)
- Security requirements
  - Non-repudiation and authentication



SECTET – Model-Based Configuration of Service Oriented Systems

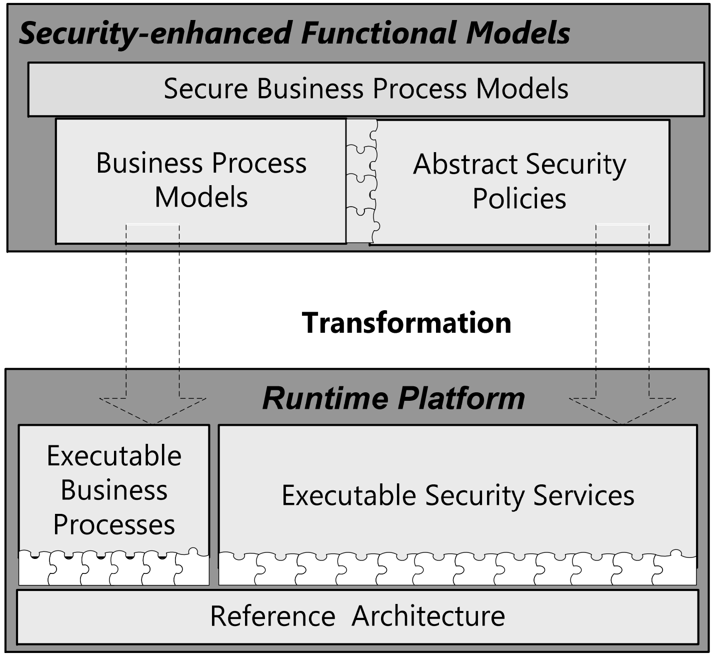


1. MDS:
Models configure services of a security architecture

2. SeAAS:
Security architecture is based on security as a service paradigm

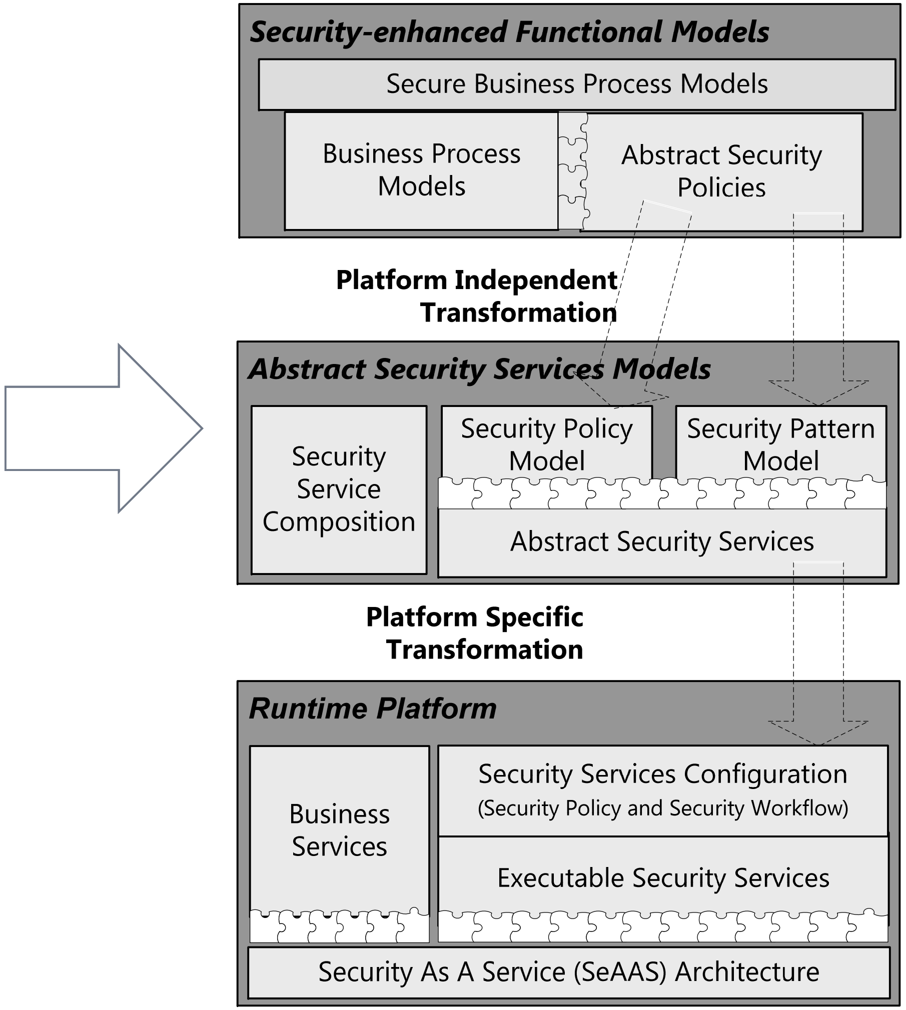
QUALITY ENGINEERING

SECTET Methodology – Model Driven Security (MDS)



Security-enhanced Functional Models
Secure Business Process Models
Business Process Models
Abstract Security Policies
Transformation
Runtime Platform
Executable Business Processes
Executable Security Services
Reference Architecture

Traditional MDS approach

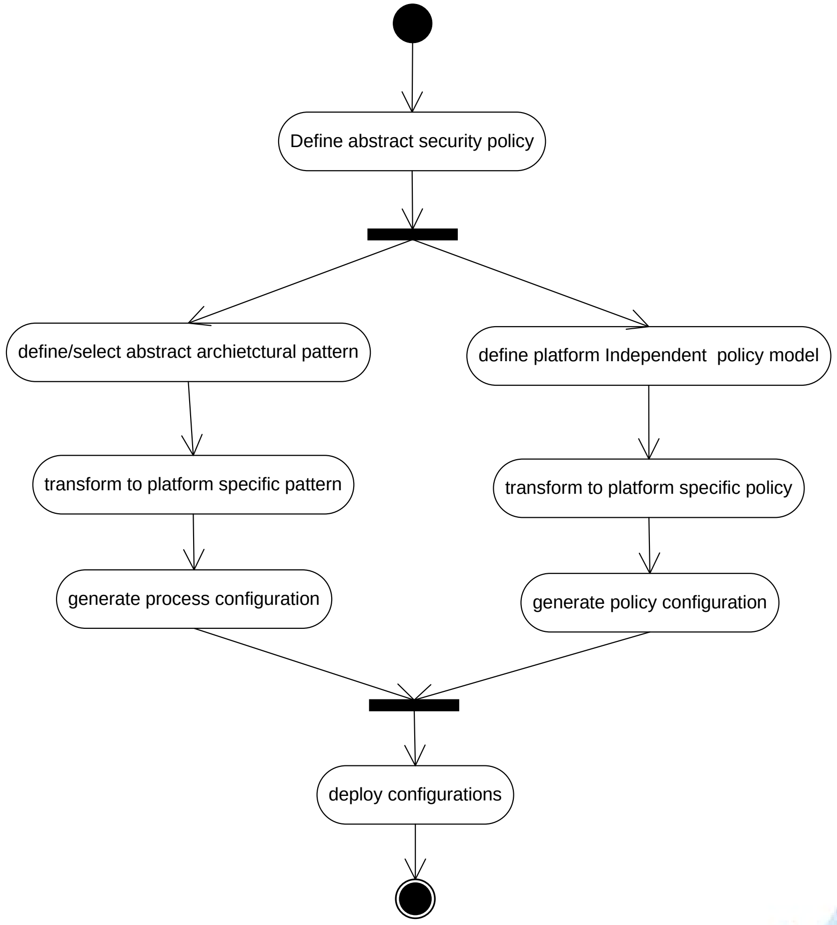


Security-enhanced Functional Models
Secure Business Process Models
Business Process Models
Abstract Security Policies
Platform Independent Transformation
Abstract Security Services Models
Security Service Composition
Security Policy Model
Security Pattern Model
Abstract Security Services
Platform Specific Transformation
Runtime Platform
Business Services
Security Services Configuration
(Security Policy and Security Workflow)
Executable Security Services
Security As A Service (SeAAS) Architecture

SECTET MDS approach

SECTET Model Driven Security Process

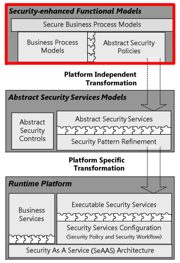
- Two procedures are considered in SECTET MDS approach
  - Architectural pattern refinement
  - Security policy model transformations
- Two artifacts are generated
  - Security policy configuration
  - Security service process configuration


Define abstract security policy
define/select abstract archietctural pattern
define platform Independent policy model
transform to platform specific pattern
transform to platform specific policy
generate process configuration
generate policy configuration
deploy configurations

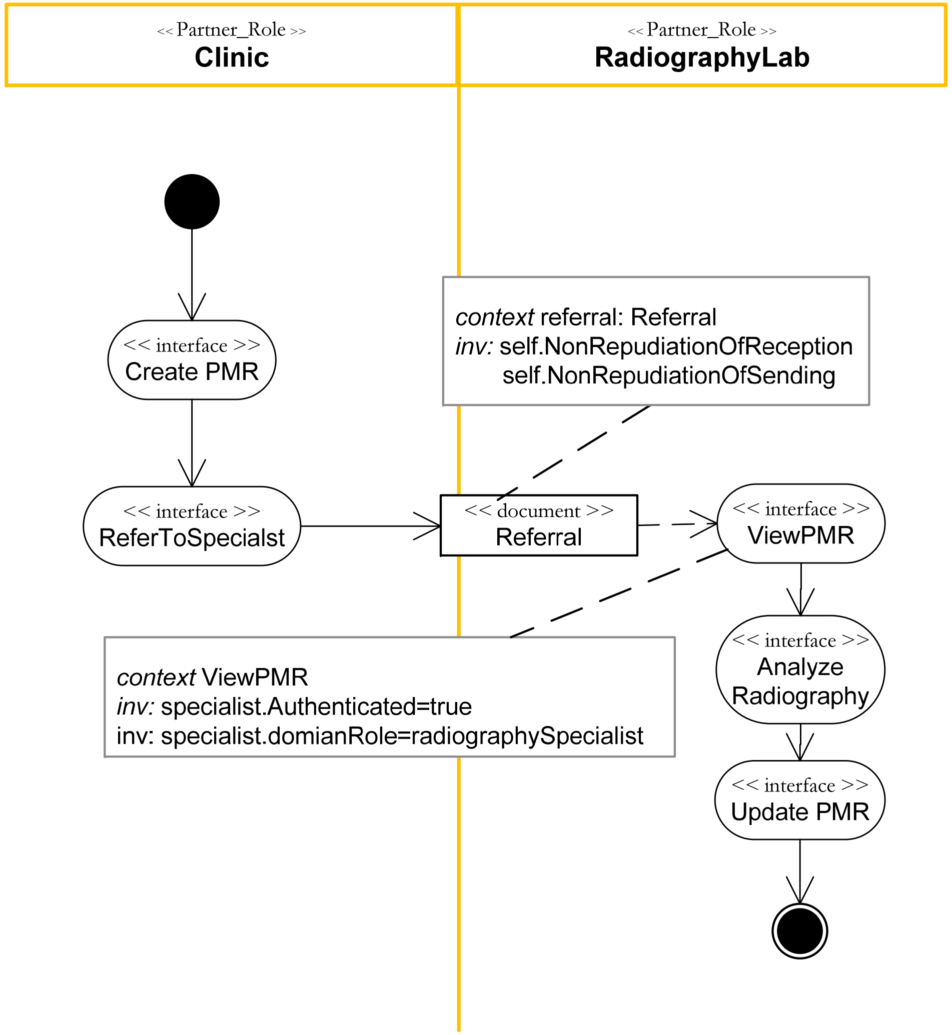
Model Driven Security (MDS) – Benefits

- Integrate security concerns in the early stage of system development
- Enrich functional models with security extensions that represent abstract security policies
- Generate declarative security policies and process configurations
- Separate tasks between: domain experts, security experts and the system administration

- *Support multiple security patterns for each requirement*
- *Enhance management and configurabilty of the architecture*

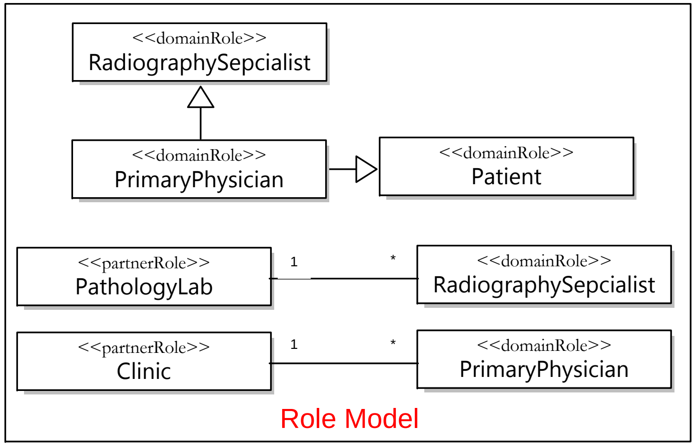

Security-enhanced Functional Models
Secure Business Process Models
Business Process Models
Abstract Security Policies
Platform Independent Transformation
Abstract Security Services Models
Abstract Security Controls
Abstract Security Services
Security Pattern Refinement
Platform Specific Transformation
Runtime Platform
Business Services
Executable Security Services
Security Services Configuration
(Security Policy and Security Workflow)
Security As A Service (SeAAS) Architecture

Security Enhanced Functional Models

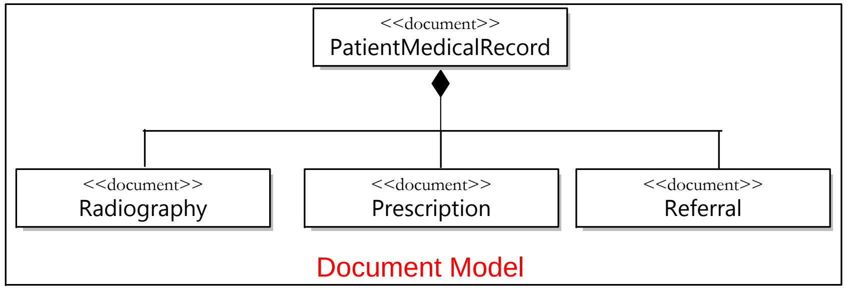


context referral: Referral
inv: self.NonRepudiationOfReception
self.NonRepudiationOfSending

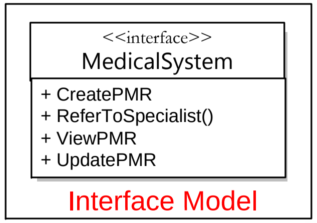
context ViewPMR
inv: specialist.Authenticated=true
inv: specialist.domianRole=radiographySpecialist



Role Model



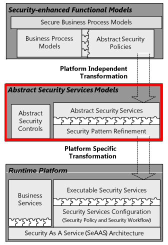
Document Model



Interface Model

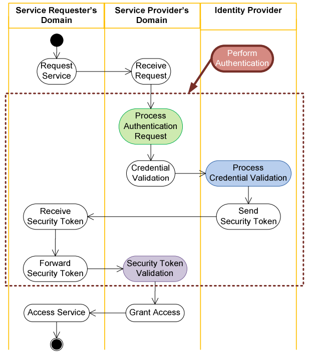
QUALITY ENGINEERING

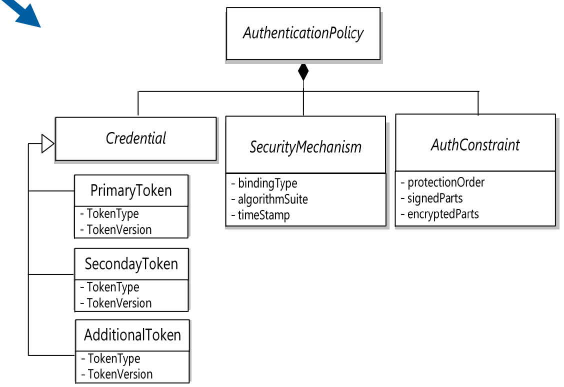
Abstract Security Models Layer


Security-enhanced Functional Models
Secure Business Process Models
Business Process Models
Abstract Security Policies
Platform Independent Transformation
Abstract Security Services Models
Abstract Security Controls
Abstract Security Services
Security Pattern Refinement
Platform Specific Transformation
Runtime Platform
Business Services
Executable Security Services
Security Services Configuration
(Security Policy and Security Workflow)
Security As A Service (SeAAS) Architecture

Abstract Authentication Policy

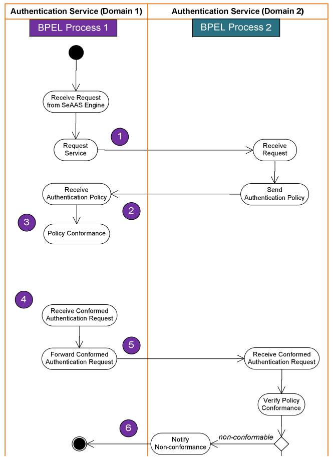
Security Architectural Patterns

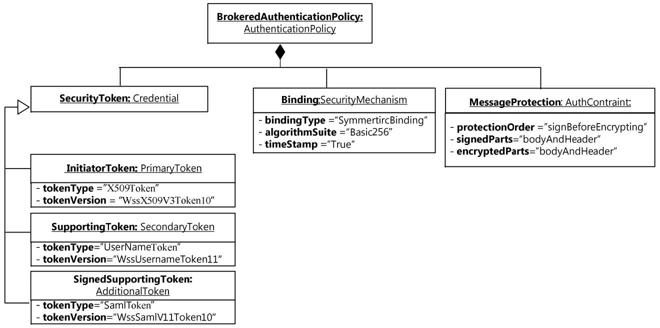




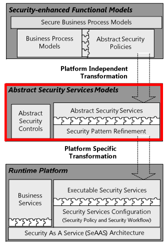
Security Policy Model

Platform Specific Architecture


Authentication Service (Domain 1)
Authentication Service (Domain 2)
BPEL Process 1
BPEL Process 2
Receive Request from SeAAS Engine
1
Request Service
Receive Request
Receive Authentication Policy
Send Authentication Policy
2
3
Policy Conformance
4
Receive Conformed Authentication Request
5
Forward Conformed Authentication Request
Receive Conformed Authentication Request
Verify Policy Conformance
6
Notify Non-conformance
non-conformable


BrokeredAuthenticationPolicy: AuthenticationPolicy
SecurityToken: Credential
Binding:SecurityMechanism
- bindingType ="SymmertircBinding"
- algorithmSuite ="Basic256"
- timeStamp ="True"
MessageProtection: AuthContraint:
- protectionOrder ="signBeforeEncrypting"
- signedParts="bodyAndHeader"
- encryptedParts="bodyAndHeader"
InitiatorToken: PrimaryToken
- tokenType ="X509Token"
- tokenVersion = "WssX509V3Token10"
SupportingToken: SecondaryToken
- tokenType="UserNameToken"
- tokenVersion="WssUsernameToken11"
SignedSupportingToken: AdditionalToken
- tokenType="SamlToken"
- tokenVersion="WssSamlV11Token10"

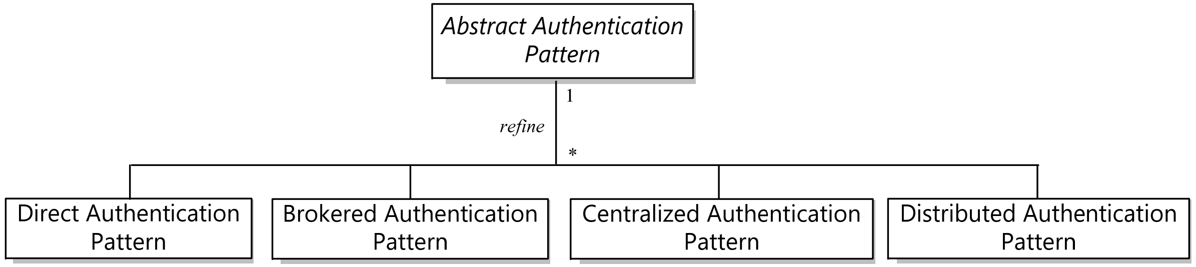
Instant Security Policy


Security-enhanced Functional Models
Secure Business Process Models
Business Process Models
Abstract Security Policies
Platform Independent Transformation
Abstract Security Services Models
Abstract Security Controls
Abstract Security Services
Security Pattern Refinement
Platform Specific Transformation
Runtime Platform
Business Services
Executable Security Services
Security Services Configuration
(Security Policy and Security Workflow)
Security As A Service (SeAAS) Architecture

Model Deriven Security – Architectural Patterns

Security Pattern Refinement Example: **Authentication**

1) **Platform-independent refinement** to security architectural pattern



Abstract Authentication Pattern

1

refine

*

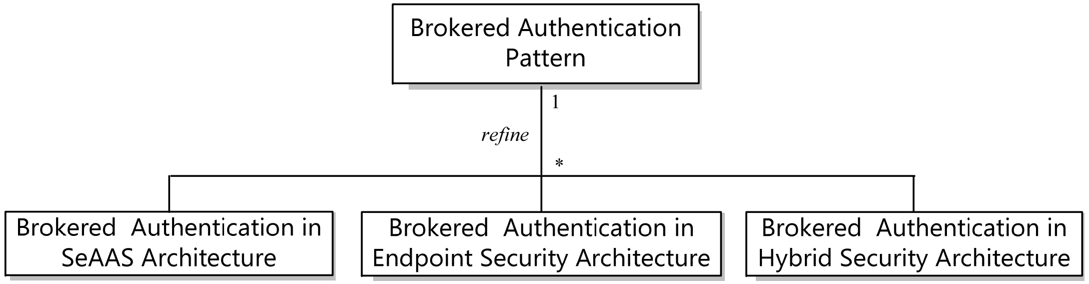
Direct Authentication Pattern

Brokered Authentication Pattern

Centralized Authentication Pattern

Distributed Authentication Pattern

2) **Platform-specific refinement** to target architecture



Brokered Authentication Pattern

1

refine

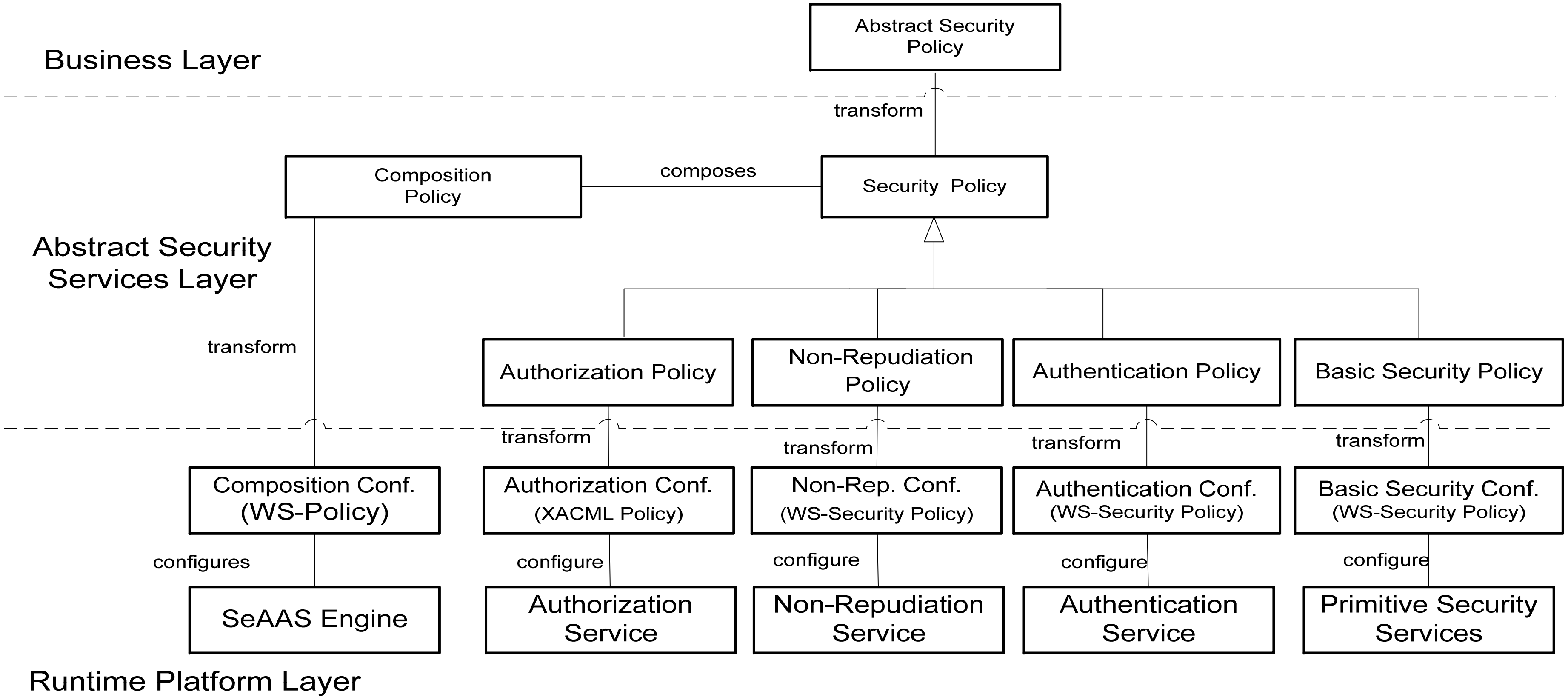
*

Brokered Authentication in SeAAS Architecture

Brokered Authentication in Endpoint Security Architecture

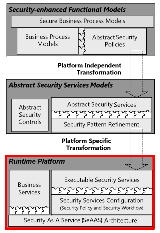
Brokered Authentication in Hybrid Security Architecture

Model Deriven Security – Security Policies



QUALITY ENGINEERING

Runtime Platform – Model Transformations

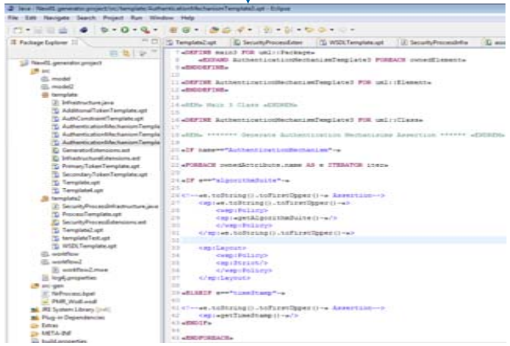

Security-enhanced Functional Models
Secure Business Process Models
Business Process Models
Abstract Security Policies
Platform Independent Transformation
Abstract Security Services Models
Abstract Security Controls
Abstract Security Services
Security Pattern Refinement
Platform Specific Transformation
Runtime Platform
Business Services
Executable Security Services
Security Services Configuration
(Security Policy and Security Workflow)
Security As A Service (SeAAS) Architecture

Source Models

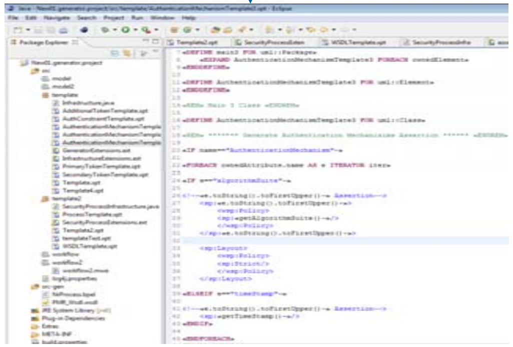
Security Policy Models

Platform-specific Pattern architecture

Transformation Templates



```
«DEFINE main3 FOR uml::Package»
    «EXPAND AuthenticationMechanismTemplate3 FOREACH ownedElement»
«ENDDEFINE»

«DEFINE AuthenticationMechanismTemplate3 FOR uml::Element»
«ENDDEFINE»

«REM» Main 3 Class «ENDREM»

«DEFINE AuthenticationMechanismTemplate3 FOR uml::Class»

«REM» ******* Generate Authentication Mechanisms Assertion ****** «ENDREM»

«IF name=="AuthenticationMechanism"-»

«FOREACH ownedAttribute.name AS e ITERATOR iter»

«IF e=="algorithmSuite"-»

<!--«e.toString().toFirstUpper()-» Assertion-->
    <sp:«e.toString().toFirstUpper()-»>
        <wsp:Policy>
        <sp:«getAlgorithmSuite()-»/>
        </wsp:Policy>
    </sp:«e.toString().toFirstUpper()-»>

    <sp:Layout>
        <wsp:Policy>
        <sp:Strict/>
        </wsp:Policy>
    </sp:Layout>

«ELSEIF e=="timeStamp"-»

<!--«e.toString().toFirstUpper()-» Assertion-->
    <sp:«getTimeStamp()-»/>
«ENDIF»

«ENDFOREACH»
```



```
«DEFINE main3 FOR uml::Package»
    «EXPAND AuthenticationMechanismTemplate3 FOREACH ownedElement»
«ENDDEFINE»

«DEFINE AuthenticationMechanismTemplate3 FOR uml::Element»
«ENDDEFINE»

«REM» Main 3 Class «ENDREM»

«DEFINE AuthenticationMechanismTemplate3 FOR uml::Class»

«REM» ******* Generate Authentication Mechanisms Assertion ****** «ENDREM»

«IF name=="AuthenticationMechanism"-»

«FOREACH ownedAttribute.name AS e ITERATOR iter»

«IF e=="algorithmSuite"-»

<!--«e.toString().toFirstUpper()-» Assertion-->
    <sp:«e.toString().toFirstUpper()-»>
        <wsp:Policy>
        <sp:«getAlgorithmSuite()-»/>
        </wsp:Policy>
    </sp:«e.toString().toFirstUpper()-»>

    <sp:Layout>
        <wsp:Policy>
        <sp:Strict/>
        </wsp:Policy>
    </sp:Layout>

«ELSEIF e=="timeStamp"-»

<!--«e.toString().toFirstUpper()-» Assertion-->
    <sp:«getTimeStamp()-»/>
«ENDIF»

«ENDFOREACH»
```

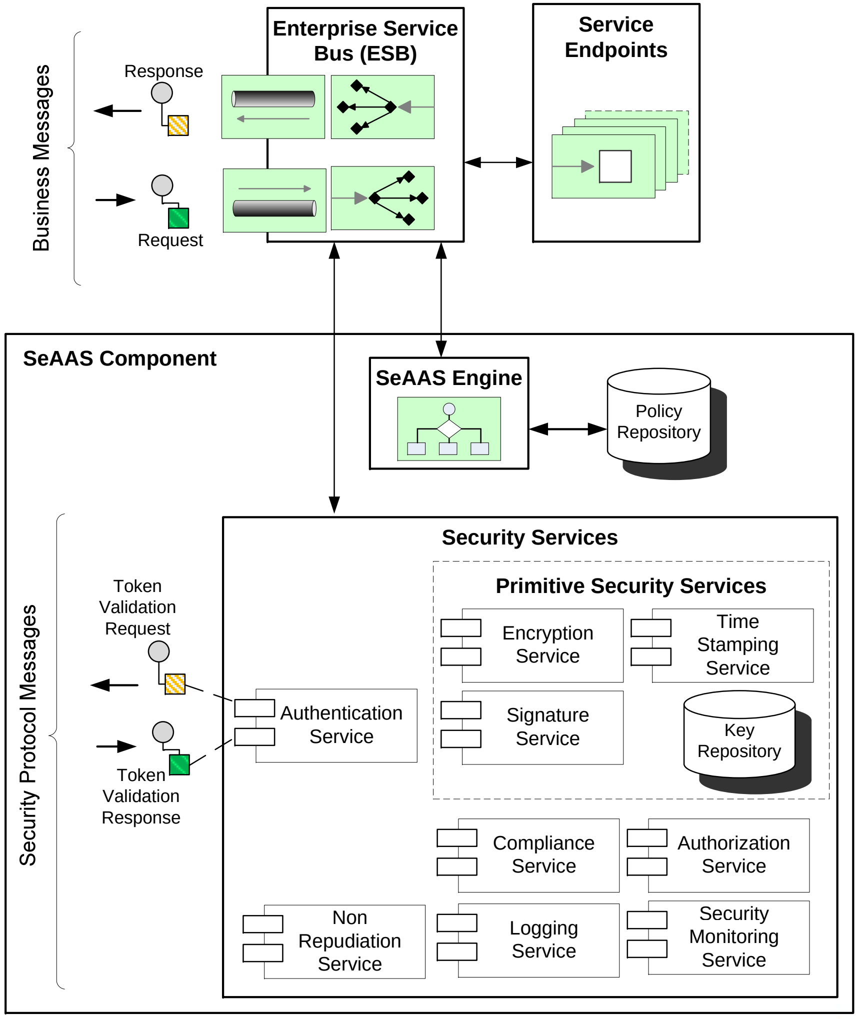
Generated Code

```
<wsp:Policy xmlns:wsp="http:// …. /policy"
<wsp:ExactlyOne>
<sp:AsymmetricBinding>
<sp:InitiatorToken>
<sp:X509Token sp:IncludeToken=".../AlwaysToRecipient">
<sp:WssX509V3Token10 />
</sp:InitiatorToken>

<sp:RecipientToken>
..
<sp:AlgorithmSuite>
<sp:TripleDesRsa15 />
...
<sp:IncludeTimestamp />

</sp:SignedEncryptedSupportingTokens>
<sp:SignedElements>
<sp:XPath xmlns:env=".../">//env:Body/*[1]</sp:XPath>
..
<sp:ContentEncryptedElements>
<sp:XPath xmlns:env="...e/">//env:Body/*[1]</sp:XPath>
</sp:ContentEncryptedElements>
</wsp:ExactlyOne>
</wsp:Policy>
```

```
<bpws:process exitOnStandardFault="yes" name="NRP" >
<bpws:partnerLinks>
 <bpws:partnerLink myRole="nro"
 name="localNROLink"
 partnerLinkType="tns:NRProcess"/>
</bpws:partnerLinks>

<bpws:invoke
 operation="requestNRO"
 partnerLink="remoteNROLink"
 portType="tns:NRO"
 inputVariable="evidenceRequest"/>

<bpws:receive
 operation="receiveNRO"
 partnerLink="localNRRLink"
 portType="tns:NRR" variable="receiveEvidence">
  </bpws:sequence>
</bpws:process>
```

SECTET Methodology – SeAAS Reference Architecture

- **Features:**
  - Dedicated shared services in a security domain
  - Decoupled from service endpoints
  - SeAAS security compositions engine
  - Out-of-bound protocol execution
  - Message oriented integration with ESB
  - WS-based Standards

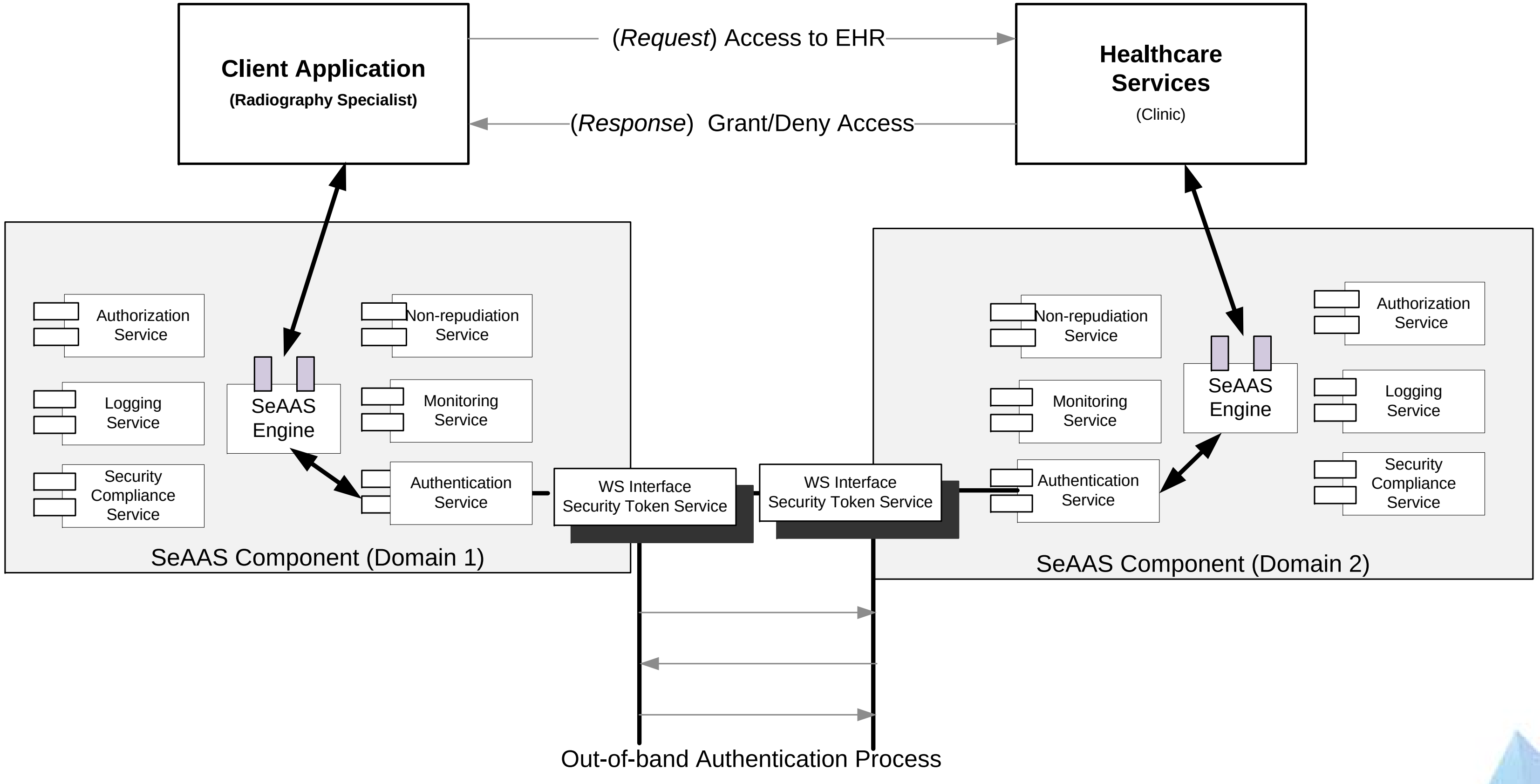


- **Benefits**
  - Better performance
  - Easy deployment/management
  - Configurable security components
  - Security service composition
  - Loosely coupled components
  - Extendable architecture

QUALITY ENGINEERING

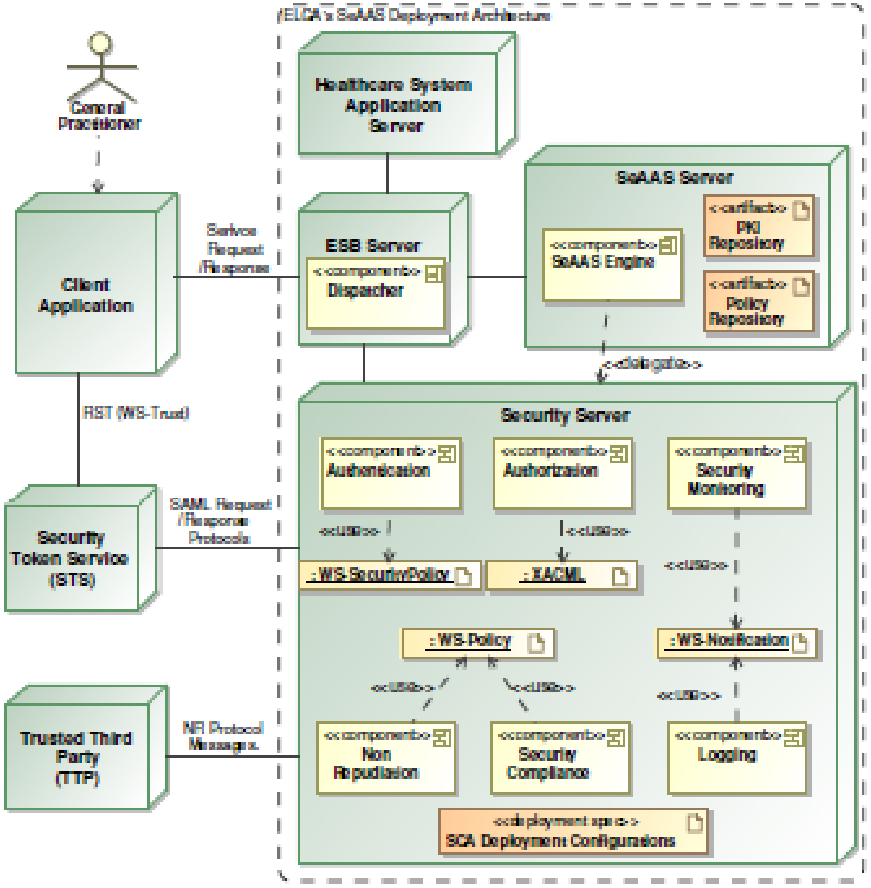
Complex Security Services Executions

- Security workflow for complex security service
- Security WS interface for Inter-Domain interactions



SECTET Methodology – SeAAS Implementation

- The delivery of security functionality over infrastructure components in a service oriented manner


ELGA's SeAAS Deployment Architecture
General Practitioner
Client Application
Service Request /Response
RST (WS-Trust)
Security Token Service (STS)
SAML Request /Response Protocols
Trusted Third Party (TTP)
NR Protocol Messages
Healthcare System Application Server
ESB Server
<<component>> Dispatcher
SeAAS Server
<<component>> SeAAS Engine
<<artifact>> PKI Repository
<<artifact>> Policy Repository
<<delegate>>
Security Server
<<component>> Authentication
<<component>> Authorization
<<component>> Security Monitoring
<<use>>
<<use>>
<<use>>
: WS-SecurityPolicy
: XACML
: WS-Policy
: WS-Notification
<<use>>
<<use>>
<<use>>
<<component>> Non Repudiation
<<component>> Security Compliance
<<component>> Logging
<<deployment spec>> SCA Deployment Configurations

SECTET – An Overview

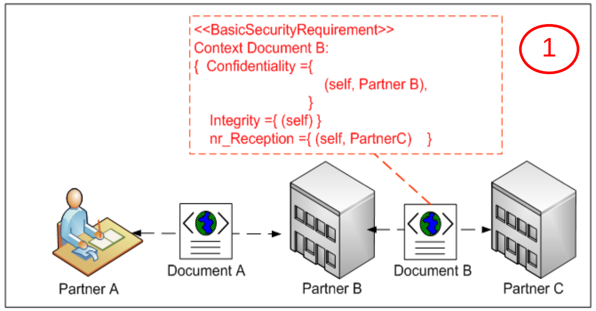
Vision

„The systemic realization of security-critical inter-organizational cooperations based on generic, composable security servcies."

Components

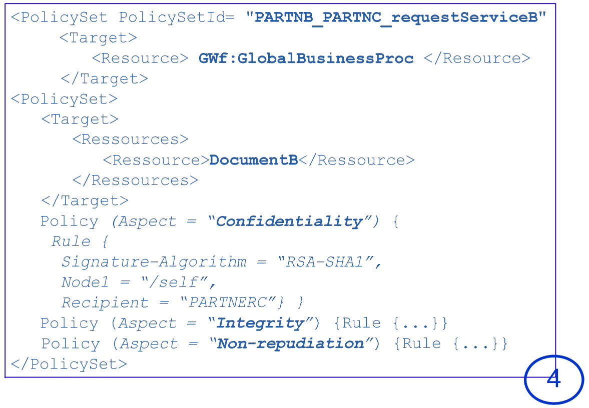
- •An extensible domain specific language
- •A reference architecture based on Security As a Service (SeAAS)
- •A multi-level transformation framework for Model Driven Security

From Platform Independent Models



<<BasicSecurityRequirement>>
Context Document B:
{ Confidentiality ={
(self, Partner B),
}
Integrity ={ (self) }
nr_Reception ={ (self, PartnerC) }

1

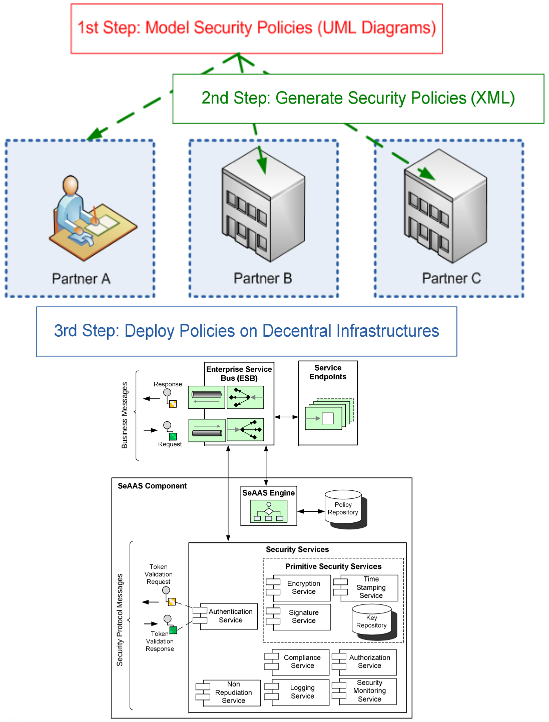


```
<PolicySet PolicySetId= "PARTNB_PARTNC_requestServiceB"
     <Target>
        <Resource> GWf:GlobalBusinessProc </Resource>
     </Target>
<PolicySet>
   <Target>
      <Ressources>
         <Ressource>DocumentB</Ressource>
      </Ressources>
   </Target>
   Policy (Aspect = "Confidentiality") {
    Rule {
     Signature-Algorithm = "RSA-SHA1",
     Nodel = "/self",
     Recipient = "PARTNERC"} }
   Policy (Aspect = "Integrity") {Rule {...}}
   Policy (Aspect = "Non-repudiation") {Rule {...}}
</PolicySet>
```

4

... to Code Artefacts

Model Driven Security Process



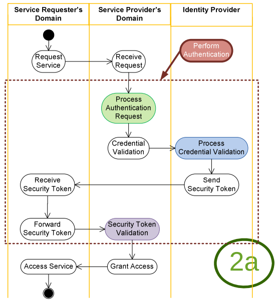
1st Step: Model Security Policies (UML Diagrams)

2nd Step: Generate Security Policies (XML)

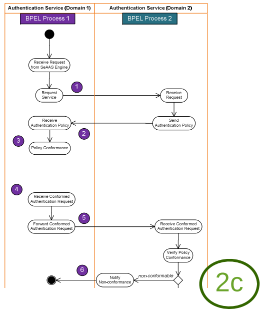
3rd Step: Deploy Policies on Decentral Infrastructures

SeAAS Infrastructure

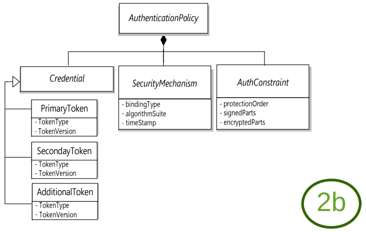
Abstract Patterns



Specific Patterns



Authentication Service (Domain 1)
Authentication Service (Domain 2)
BPEL Process 1
BPEL Process 2
Receive Request from SeAAS Engine
1
Request Service
Receive Request
Receive Authentication Policy
Send Authentication Policy
2
3
Policy Conformance
4
Receive Conformed Authentication Request
5
Forward Conformed Authentication Request
Receive Conformed Authentication Request
Verify Policy Conformance
6
Notify Non-conformance
non-conformable
2c



Security Policy

Conclusion

- Collaborative systems based on SOA are heterogeneous, agile and dynamically evolving
- The best practice for SOA security is based on
  - Endpoint security
  - Traditional MDS approach to close the business-code gap is
    - Applied in one step
    - Inflexible and supports one security pattern

- Proposed SECTET framework is based on two main concepts
  - SeAAS methodology for the design of the reference architecture (RA)
  - Enhanced MDS methodology for the configuration of security services

Future Work

- Investigating further security services like security monitoring, identity management, and usage control

- Developing the formal foundation of the refinement process and security composition

- Deploying and testing an EHR system developed by our industrial partner, ITHicoserve

… Thank you for your attention!

www. sectissimo.info