

An Event-Driven Framework for Service-Oriented Computing

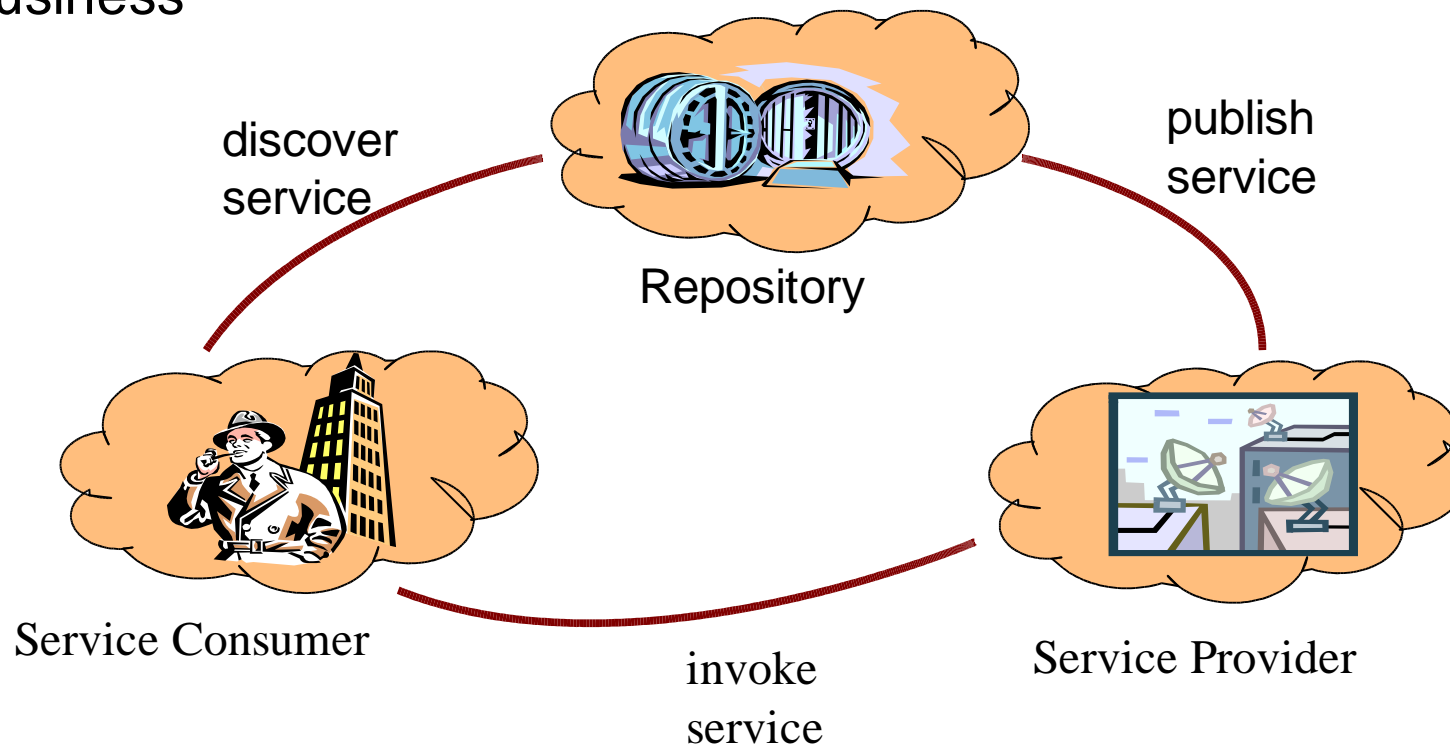
Kees Leune
March 16, 2004

Presentation Outline

- Service Oriented Computing and interoperability
- Presentation focus: loose couplings and security
- The Framework
- Future Research

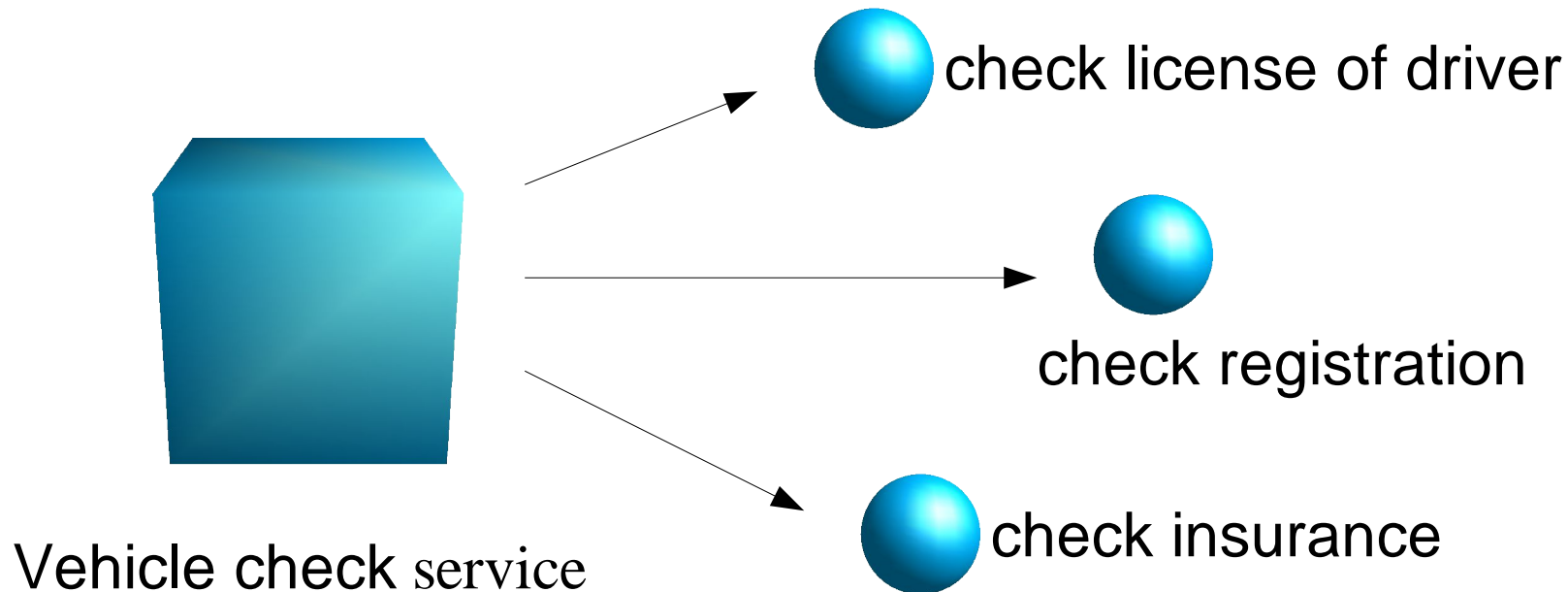
Service Oriented Computing

- Service-Oriented Computing (SOC) is a new paradigm to design, develop and implement distributed services.
- SOC is much more than technology; it is a new way of conducting business



The Concept of a Service

- *A service provides a group of related capabilities to its users.*
- *A service is not a technological concept!*



Service-Oriented Architecture

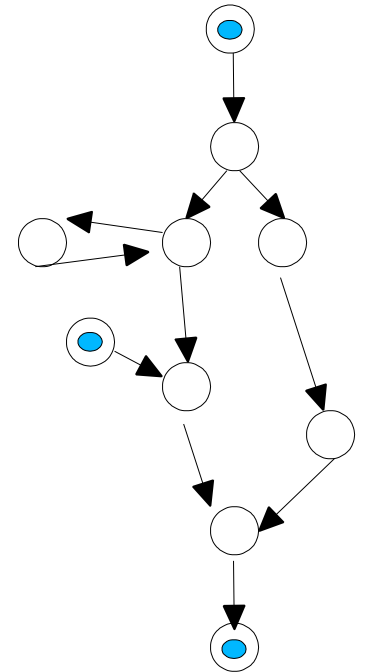
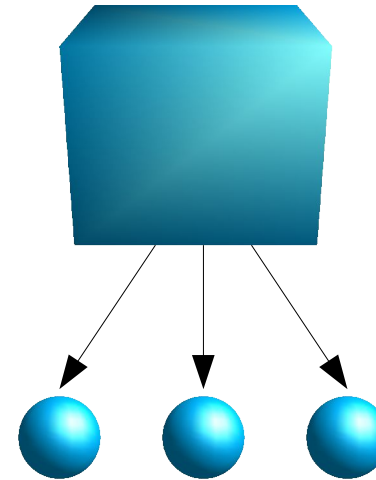
- The Service-Oriented Architecture provides the technological support for the SOC paradigm.
- Service are described using WSDL (Web-Service Description Language)
 - Messages technology-neutral description of I/O messages
 - Operations technology-neutral description of capabilities
 - Interface technology-neutral grouping of operations
 - Binding translation from technology-neutral to implementation
 - Service a reference to a (partial) implementation of an interface

Service-Oriented Architecture

- The Service-Oriented Architecture provides the technological support for the SOC paradigm.
- Service are described using WSDL (Web-Service Description Language)
 - **Messages** technology-neutral description of I/O messages
 - **Operations** technology-neutral description of capabilities
 - **Interface** technology-neutral grouping of operations
 - **Binding** translation from technology-neutral to implementation
 - **Service** a reference to a (partial) implementation of an interface

Interoperability

- SOC is a paradigm for interoperability
 - process vs. service
- Ad-hoc discovery and invocation
- Location independent
- Loosely coupled



Loose Couplings

- Service-Oriented Computing adopts a principle of ad-hoc service discovery and invocation.
 - Highly dynamic: no time for extended negotiations for exchange format (e.g. Edifact is unusable).
 - Heterogeneous environments: business use different technological platforms, different protocols, different requirements, etc.
 - Technological limitations; such as interrupted communications networks

The need for security

Assume a basic SOC scenario:

- Our organization needs a specific capability for one of our business processes and we discover service A
- Company X provides service A
- Our organization does not know anything about company X
- To effectively use service A, it must be allowed to interact with our business processes
- Since company X is not known, it cannot be trusted
- Since company X is not trusted, it cannot interact with our business processes
- Since company X cannot interact with our business processes, service A cannot be used
- Since service A cannot be used, we cannot provide our business process

Trust

Trust is the extent to which a party exhibits expected behavior.

Trust may be increased by

- Limiting the ability to exhibit unexpected behavior.

Quality aspects of security

- Authentication
 - Establishing the identify of a subject
- Authorization
 - Establishing the permissions of a subject
- Confidentiality
 - Ensuring that messages may only be seen by authorized subjects
- Integrity
 - Ensuring that messages can only be modified by authroized subjects
- Non-repudiation
 - Ensuring that sending or receiving of messages cannot be denied.

Research Goal

To develop a secure framework for event-driven service-oriented computing and show its validity by implementing a prototype and testing it in a case study environment.

- Secure To provide an acceptable level of trust
- Event-driven To enable loose couplings between processes and services

Focus on integrating services in existing business processes, rather than on creating and/or publishing new services.

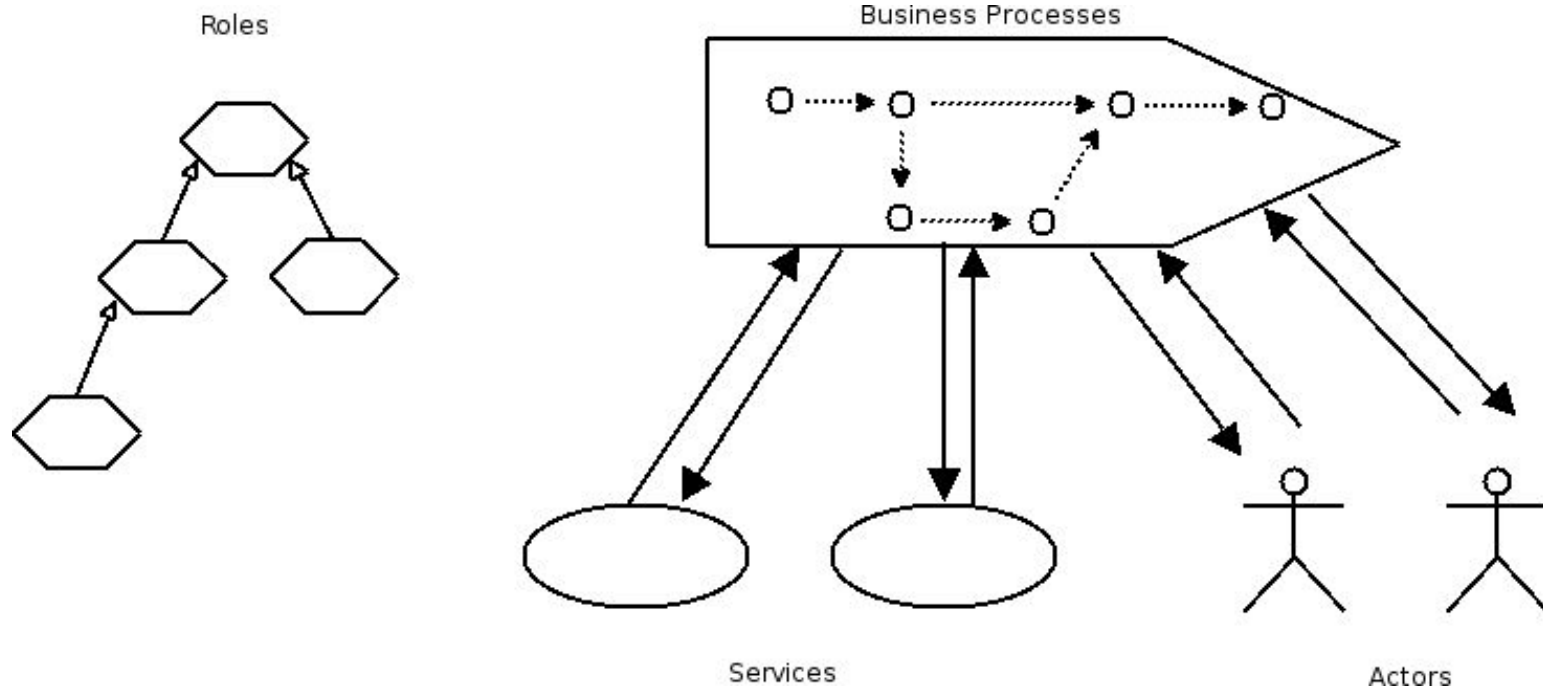
The EFSOC Framework

The EFSOC framework consists of three tiers:

- *Event Tier*
All communication between subjects, such as actors, services or processes, takes place via event exchange
- *Security Tier*
All communications must be secure
- *Business Process Tier*
All communications are driven by business requirements

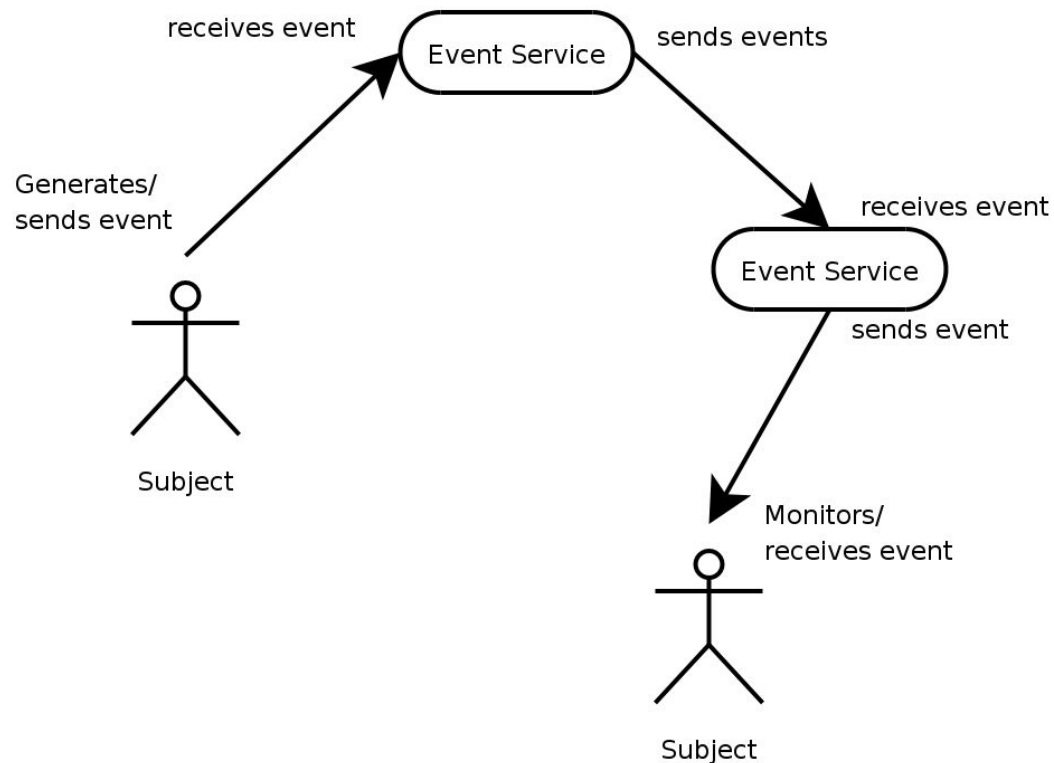
Each tier is supported by an infrastructure service.

The EFSOC Framework



Event Tier

- Events represent “things” happening in organizations.
- Events and are sent and received by subjects.

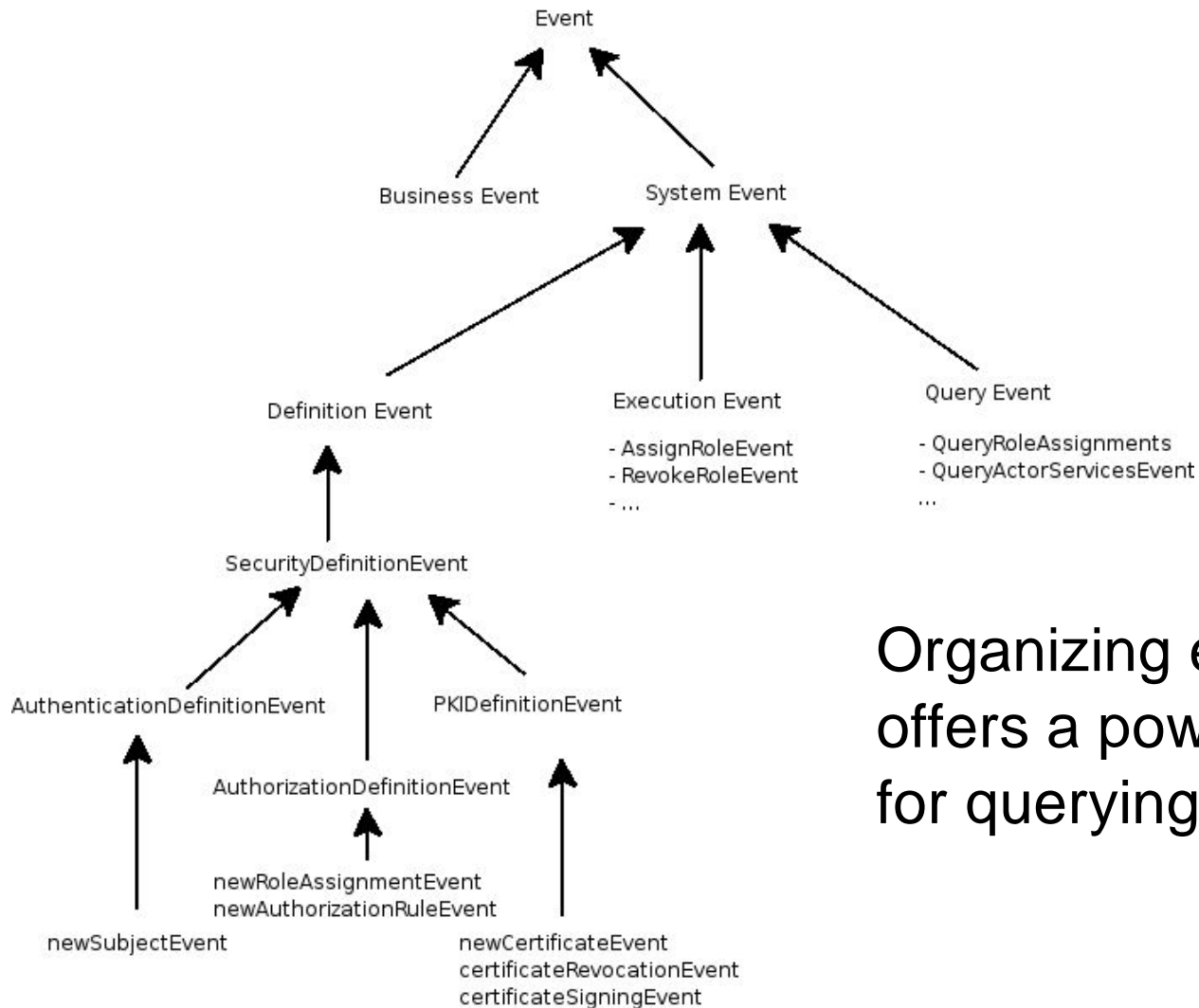


Events

- Events have an envelope and a body
 - The event header contains meta-information used for security and routing
 - The event body contains additional information.
- All event body types must be defined before they are sent.
- A subject needs to subscribe to an event type before he receives it.

```
<event>
  <envelope>
    <eventtype>LoanRequestEvent</eventtype>
    <eventid>20040204193</eventid>
    <timestamp>1075890192</timestamp>
    <sender>john</sender>
  </envelope>
  <body type="loanRequestBody">
    <interestrate>0.05</interestrate>
    <paybackperiod>12</paybackperiod>
    <loanamount>10000</loanamount>
  </body>
</event>
```

Event hierarchy



Organizing events in a hierarchy offers a powerful mechanism for querying the state of the model.

Security tier

Goal: To provide a security layer for event relay

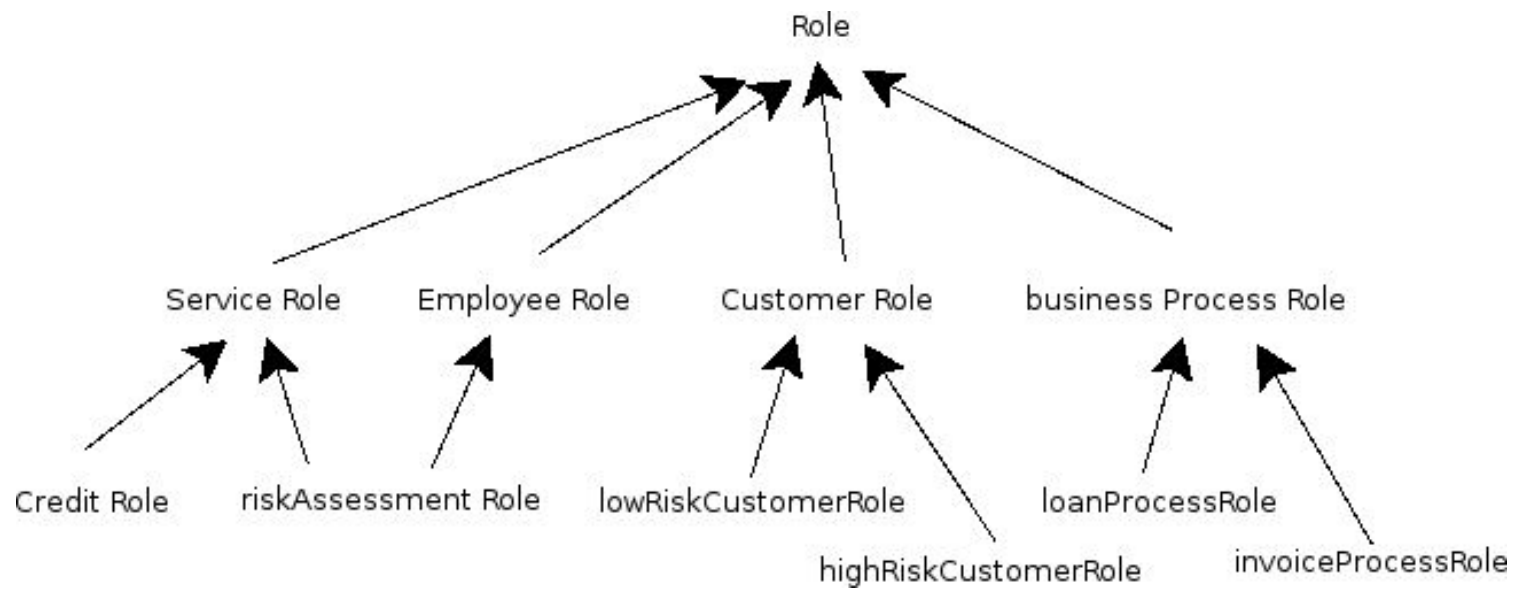
Authentication and Access control

- DAC and MAC
 - Not usable in its current form due to high management overhead.
- RBAC
 - Limited in its purest form because of lack of granularity in the *permission* concept and not well suitable for dynamic evaluation.

Security Tier

- The EFSOC security model is based on Role-Based Access Control, i.e.
 - Subjects play roles
 - Permissions are the result of dynamically evaluation authorization rules
 - Authorization rules are role-based
- Authentication:
 - unauthenticated: subject is unknown
 - partially authenticated: subject is known, but plays no roles
 - fully authenticated: subject is known and plays at least one role.

Example Role Hierarchy



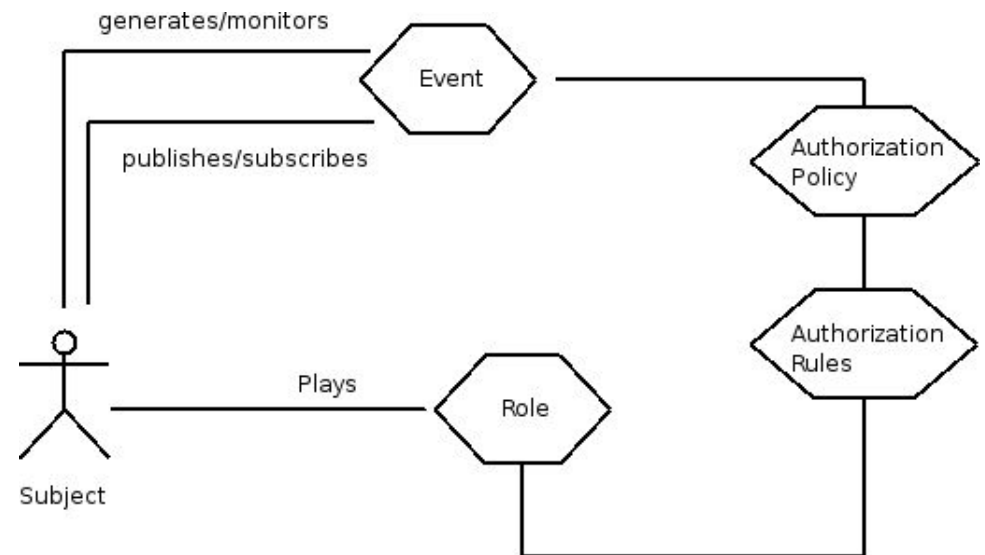
Security tier

Authorization

- Associated with each Event Type is an Authorization Policy containing one or more rules.

Confidentiality and Integrity

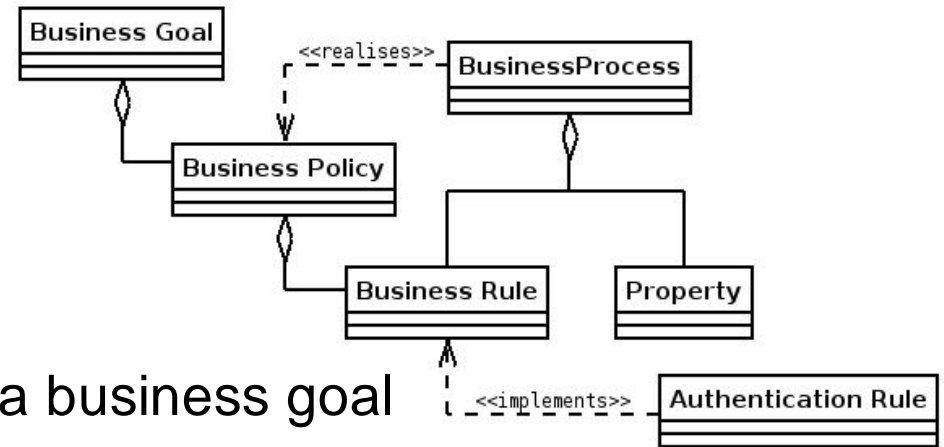
- Are achieved by using well-known public key infrastructure techniques



Business Process Tier

Describes business protocols and service specifications.

- A business process implements a business policy, which is driven by a business goal
- A business process may be perceived as a collection of business rules.
- A subject of the business rules will be implemented as authorization rules



Process specification

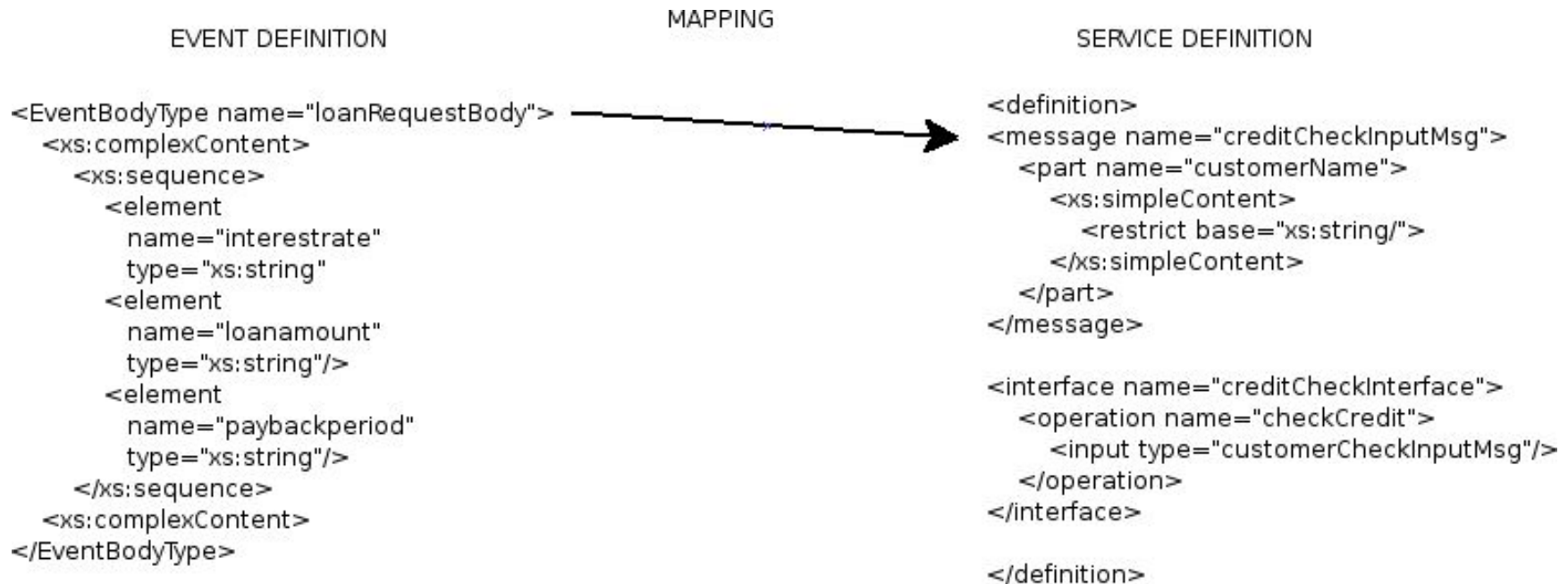
- Only interoperability aspects of business processes have to be represented.

Example

```
<bpcontext name="loancontext">
  <bprocess name="loanprocess">
    <use rule="R01"/>
    <use rule="R02"/>
    <use rule="R03"/>
    <binding rule="R03" type="soapbinding"/>
    <monitors eventbodytype="loanRequestEvent"/>
    <generates eventbodytype="loanResponseEvent"/>
    <generates eventbodytype="creditCheckRequest"/>
    <generates eventbodytype="riskAssessmentRequest"/>
  </bprocess>
</bpcontext>
```

Service Specification

- Services are represented as WSDL documents. A mapping from WSDL message to EFSOC event type must be provided.



Event-Service mapping

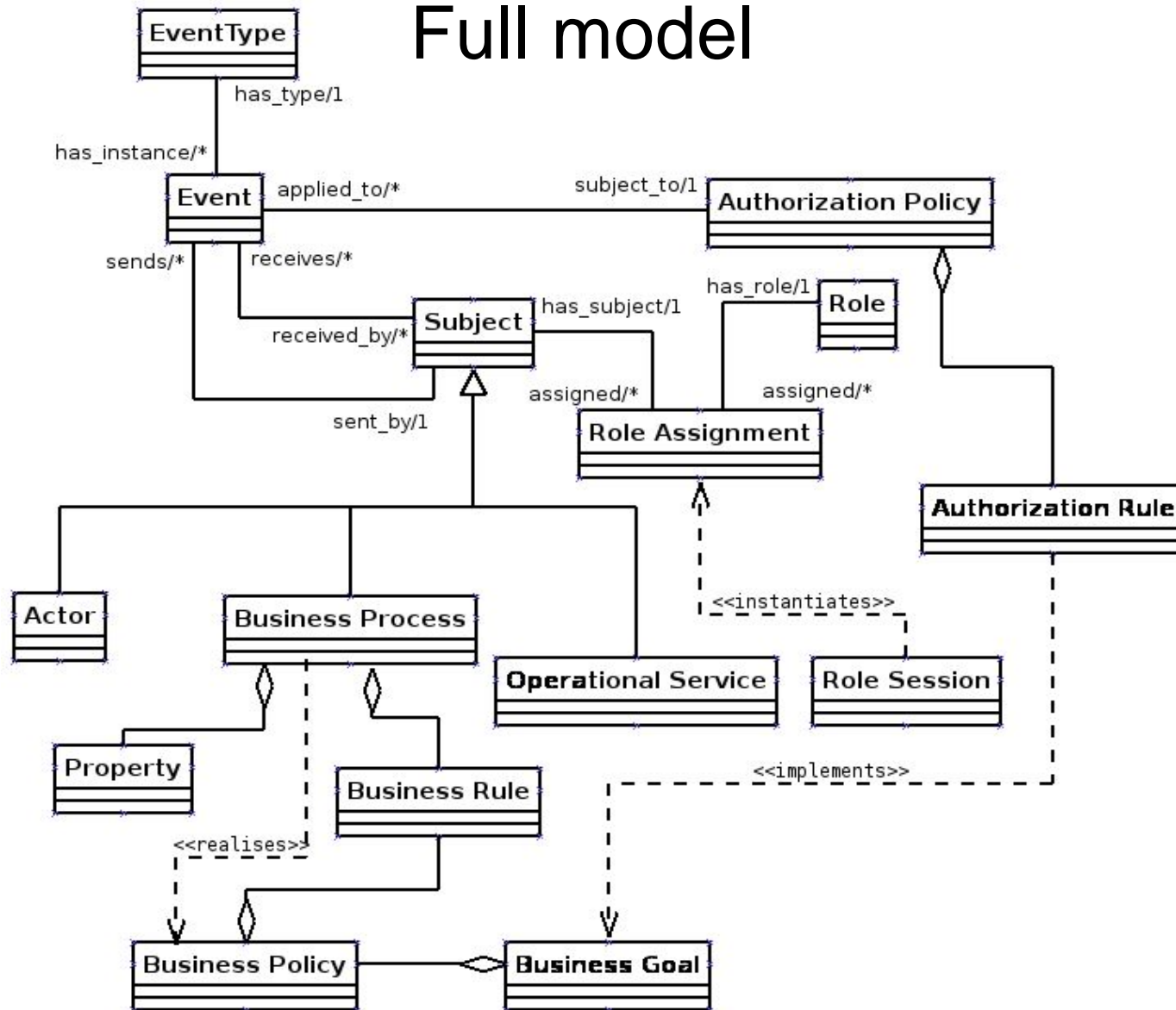
- Example

```
<servicedescription name="creditservicedescr" />
  <wsdl>http://pronir.nl/efsoc/credit.wsdl</wsdl>
  <mappings>
    <map eventbodyelement="creditcheckrequest "
        servicemessage="requestMessage" />
    <map eventbodyelement="creditcheckresponse "
        servicemessage="responsemessage" />
  </mappings>
  <invokers>
    <invoke eventbodytype="creditcheckrequest "
        operation="checkCredit" />
  </invokers>
</servicedescription>
```

Summary

- The Event-Driven Framework for Service-Oriented Computing (EFSOC) consists of three tiers
 - Event Tier: in which all communication between subjects is addressed in an event-driven fashion;
 - Security Tier: which provides a security layer to provide an acceptable level of trust;
 - Business Process Tier: which provides interoperability between the various subject types.

Full model



Future work

- Develop formal framework representation
- Query language
- Develop functioning prototype for specification and querying
- Integration with VIMES prototype (Nijmegen University)
- Case Study

Contact me

email : kees@uvt.nl
phone : +31 13 466 2688
web : <http://www.leune.org>
address : Tilburg University
Infolab, Room B738
P.O. Box 90153
5000 LE Tilburg
NETHERLANDS



Or, first door to you right when you leave