Producing Web Services using UML and MDA

Enterprise
Collaboration
Architecture

EDOC

A tutorial on applying Model Driven Architecture to web services using the OMG Enterprise Collaboration Architecture with UML

Introductions

DataAccessTechnologies

Where Business Meets Technology

Cory Casanave

cory-c@enterprise-component.com

Primary author of “CCA” in EDOC

What is the Enterprise Collaboration Architecture?

- ECA is a “profile of UML”, a way to use UML for a specific purpose - it is an OMG standard
  - That purpose is *modeling enterprise systems*.
- You can also think of this as a “modeling framework” for enterprise computing
- ECA is part of the “Model Driven Architecture” (MDA) initiative of the OMG
  - Using precise modeling techniques as part of the development lifecycle to speed development and provide technology independence
- ECA has been adopted by the OMG as part of the EDOC RFP.

Collaboration is Key

- Collaboration is a key differentiation and key cost center (Healthcare Example)
  - Customer Collaboration
  - Claim processing
  - Disputes
  - Physician Collaboration
  - Payer Collaboration
  - Hospital Collaboration
  - Broker Collaboration
  - Government Collaboration
  - Employee Collaboration
  - Others...

The Agile Enterprise

Has a competitive advantage in its capability to embrace collaboration and change

Integrating Enterprises, People & Systems - Worldwide

- Business Requirements
  - Virtual Enterprises
  - Enterprise Integration (EAI)
  - Supply-chain automation (B2B)
  - Customer Integration (B2B)
  - Web deployment (B2C)
  - Internet Marketplace (B2C)

- Collaboration and Integration

The dynamic reality

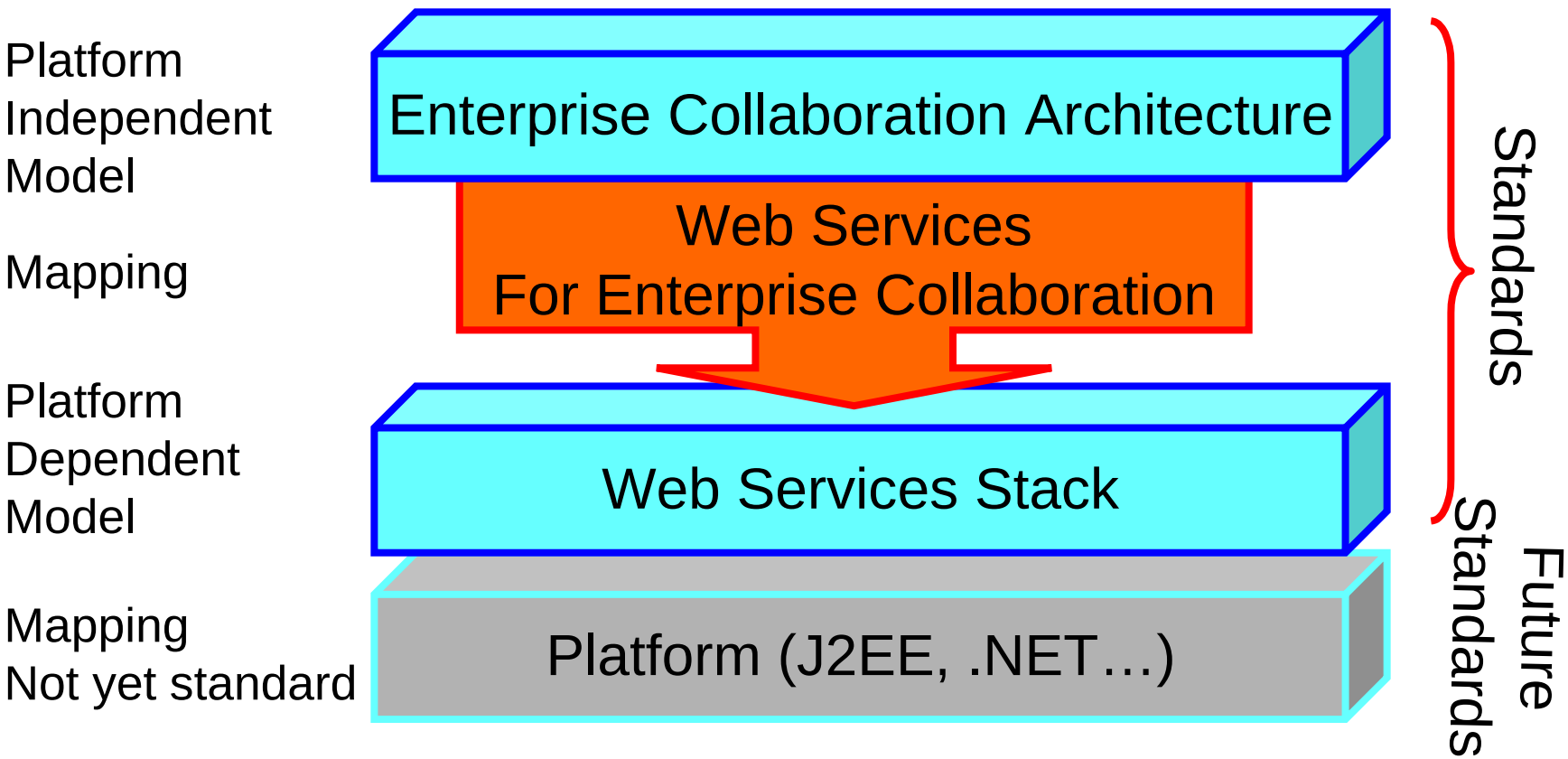
- The information system must facilitate;
  - Rapid realization of business goals
  - Integration of independent processes and systems
  - Multiple and Changing
    - *business requirements*
    - *business processes*
    - *technologies*
    - *standards*
    - *enterprise boundaries*
    - *partners*

Technology Stew

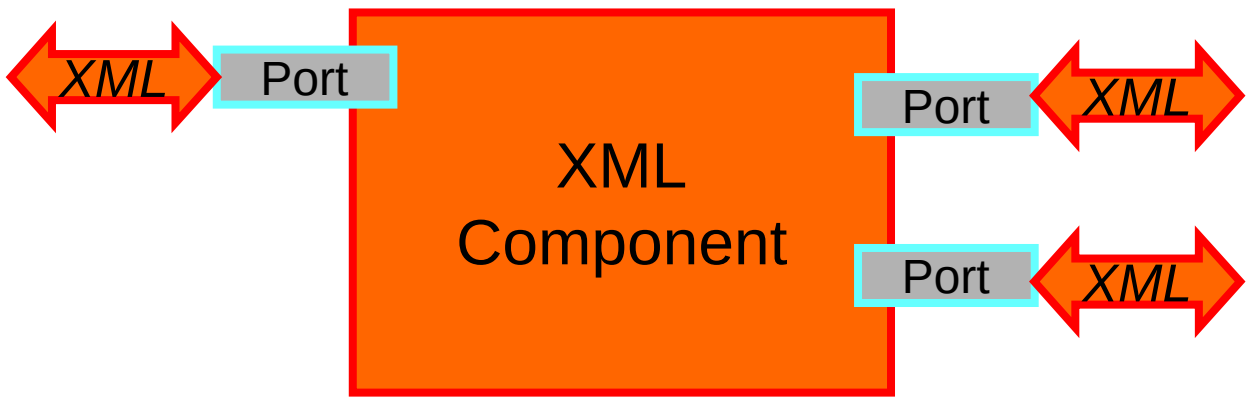
- Web services
- .NET
- C’
- XML
- EAI
- Active Web pages
- EJB
- Java Beans
- Java
- Corba
- MQ-Series
- C++
- SQL
- Cobol
- IMS
- CICS
- ...

Technology is transient, but we must embrace and adapt to it to provide meet current requirements

MDA Solution for Web Services



XML Components



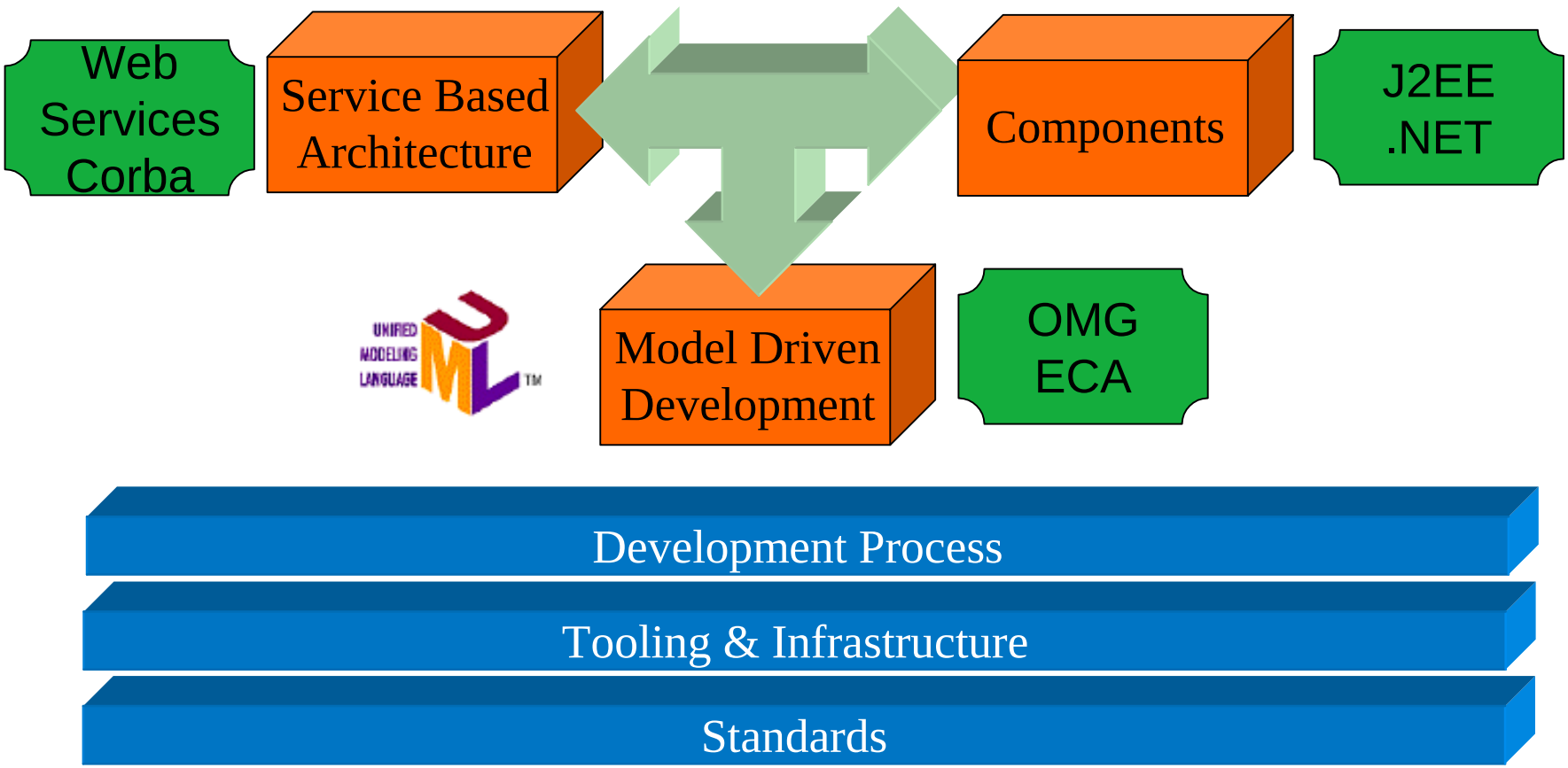
ECA and Web services together provide
An XML component architecture
Independent of protocol and platform

Problem Space

- Integration Nightmare
- Infrastructure, Version & Vendor lock-in
- Complex, divergent and manual development and deployment processes

- *Typical solutions require buy-in (Lock-in) to expensive, pervasive and proprietary infrastructure*

Solution Triad

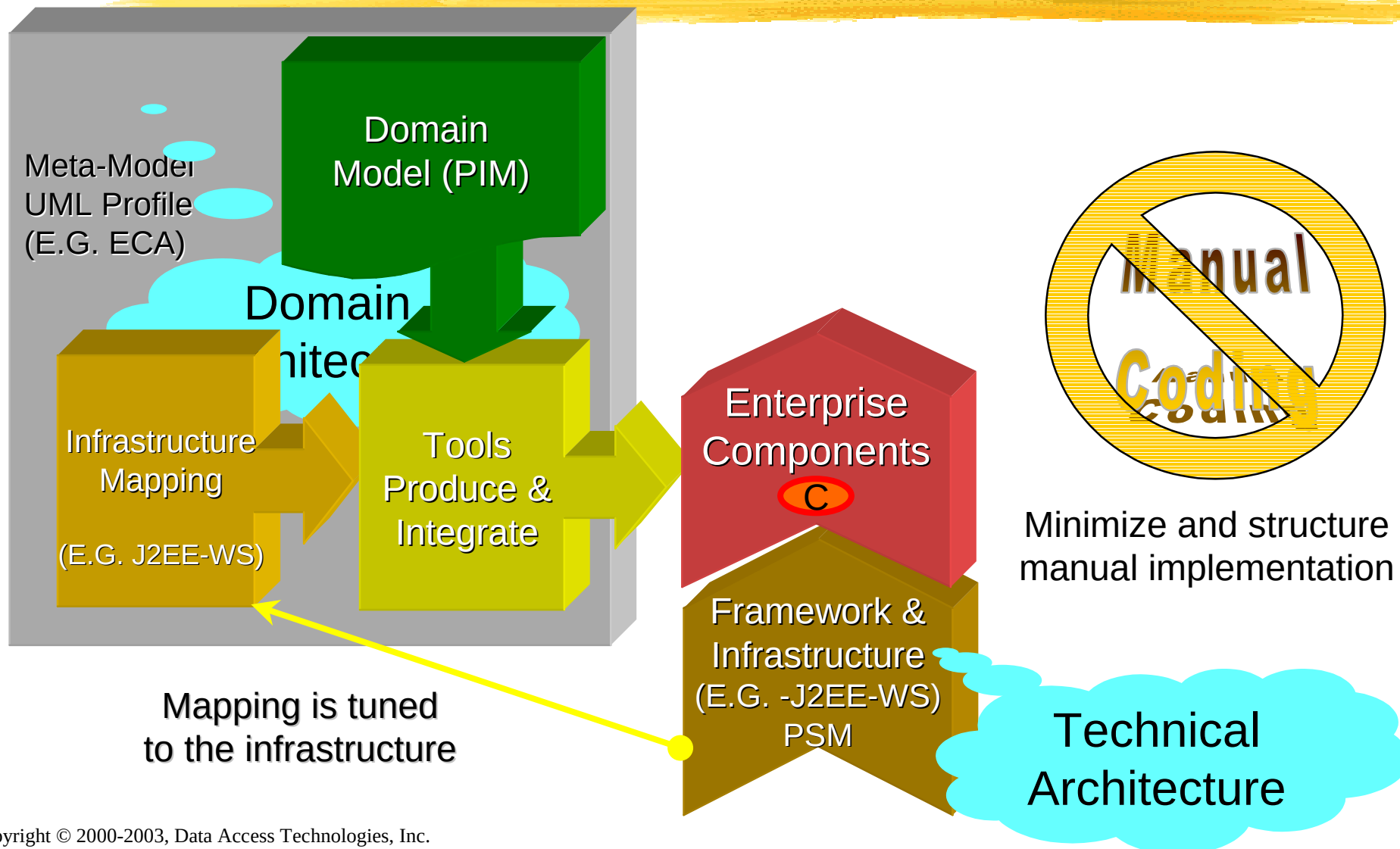


The new center

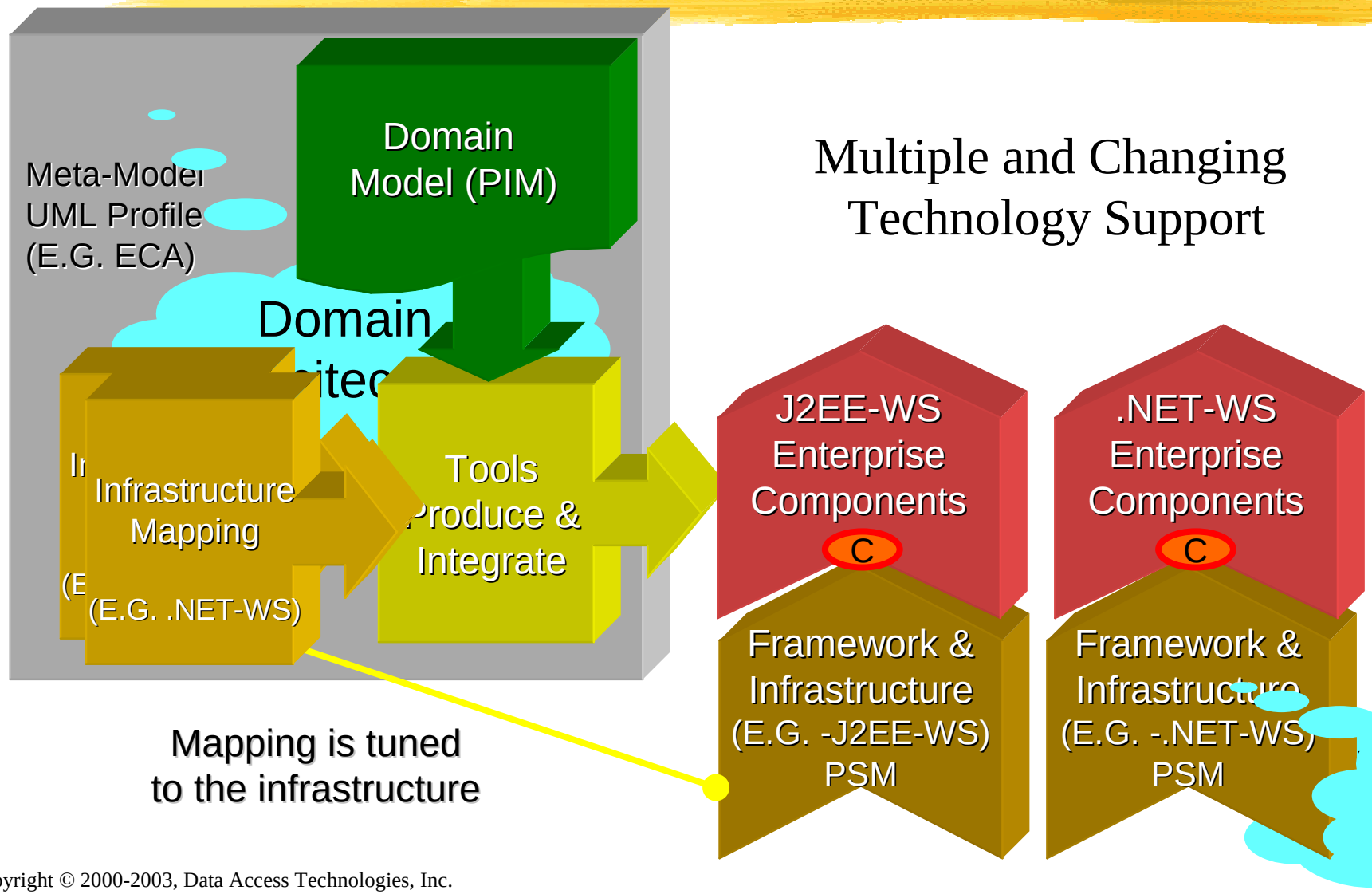
- The strategic core of you systems must be the business its self
- Only technology independent business focused models will survive the transience of technology and lock-in
- These models can become *part of your source code*, driving enterprise applications
- Enabler: Model Driven Architecture (MDA) with EDOC-ECA

Extreme Modeling

Automated Model Driven Architecture



Automated Model Driven Architecture



Enterprise Architecture

How to slice and dice

Loose Coupling

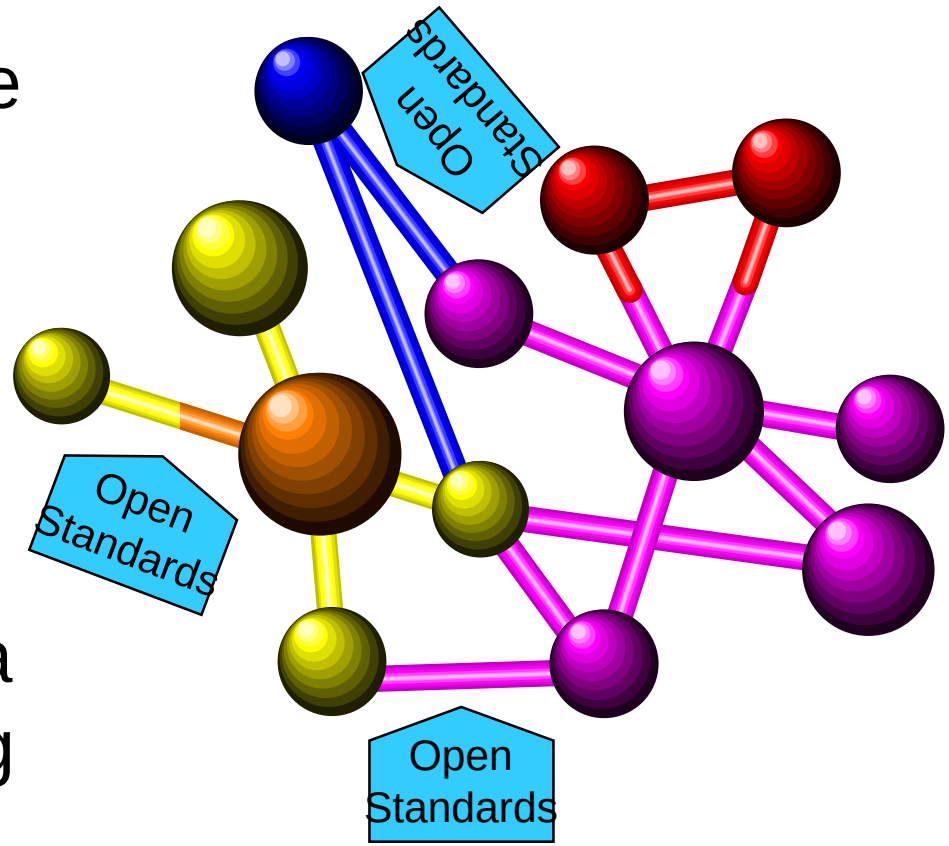
- Loose coupling is the ability for independent parts of systems to be built and evolve independently
- Tightly coupled systems
  - Prevent change (the next legacy system)
  - Cause lock-in
  - Become unmanageable
  - Prevent reuse
- Quality architecture is essential for loose coupling

Architecture Goals

- Create a system from loosely coupled “enterprise components” that can evolve independently
- Provide well defined interfaces and interaction points between these enterprise components
- Make each enterprise component a reusable asset that can serve many business processes
- Build the information system as a community of interacting enterprise components
- Utilize open standards such as XML, EJB and Corba to integrate the enterprise components

Enterprise Components

- Enterprise Components must be independent
- While being able to interoperate with each other
- Making the information system a lattice of cooperating components

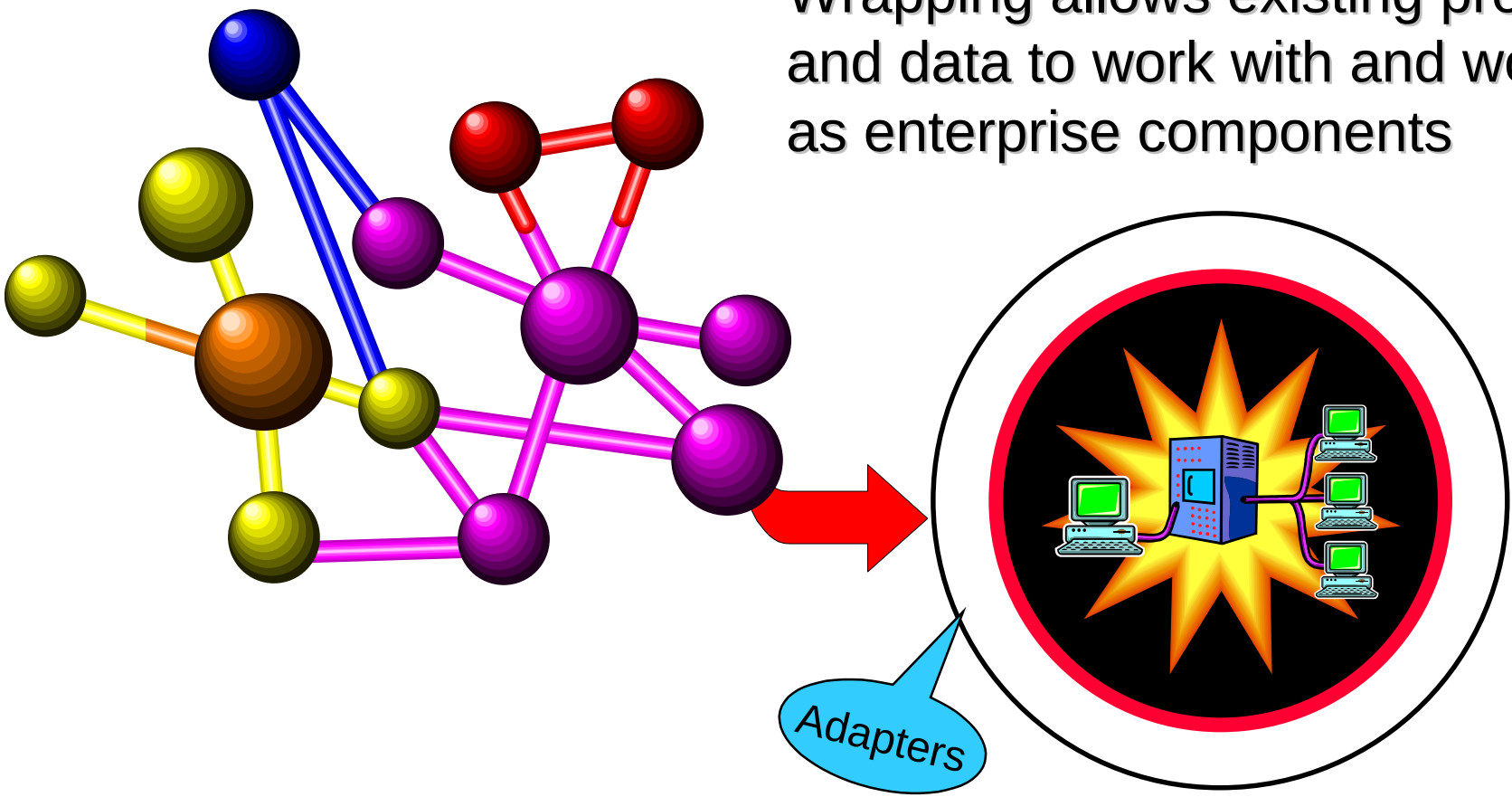


“Wrapping” Legacy Applications and Data

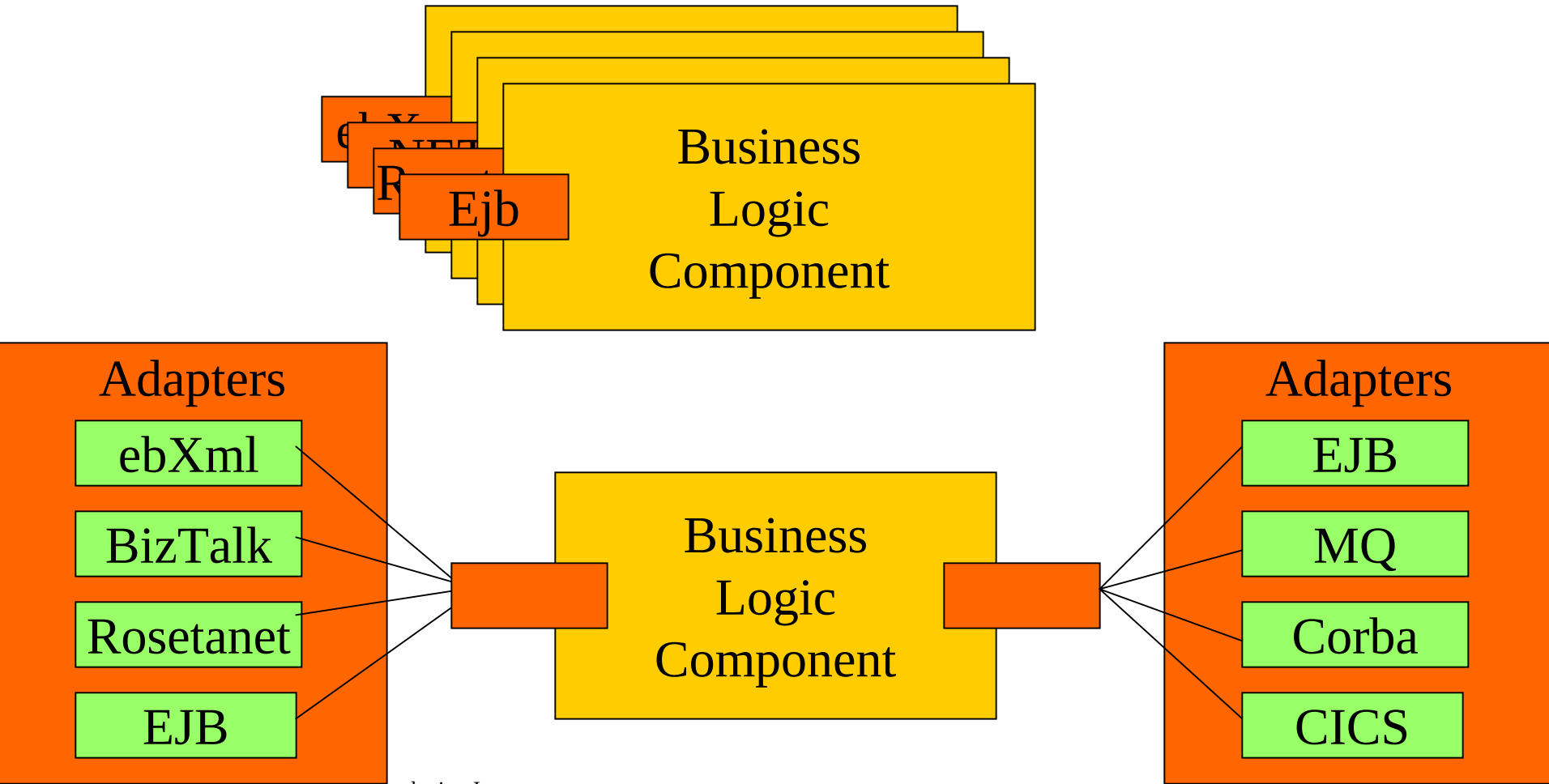
- Enterprise Components are defined in terms of their external contract; implementation may use existing applications
  - Can “call” existing application
  - Can read and write legacy DBMS
  - Can use “screen scraper” (Last resort)
- Legacy applications can appear as enterprise components but may not be implemented as components

Legacy “Wrapping”

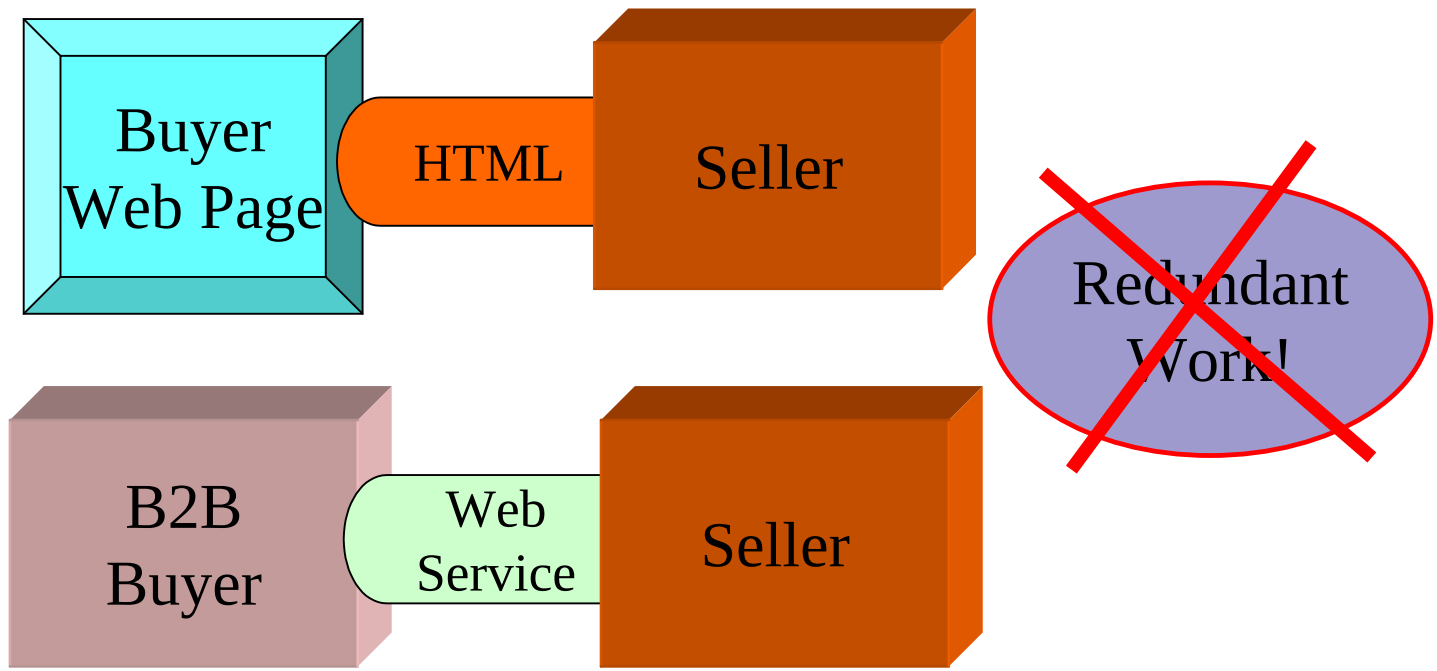
Wrapping allows existing programs and data to work with and work as enterprise components



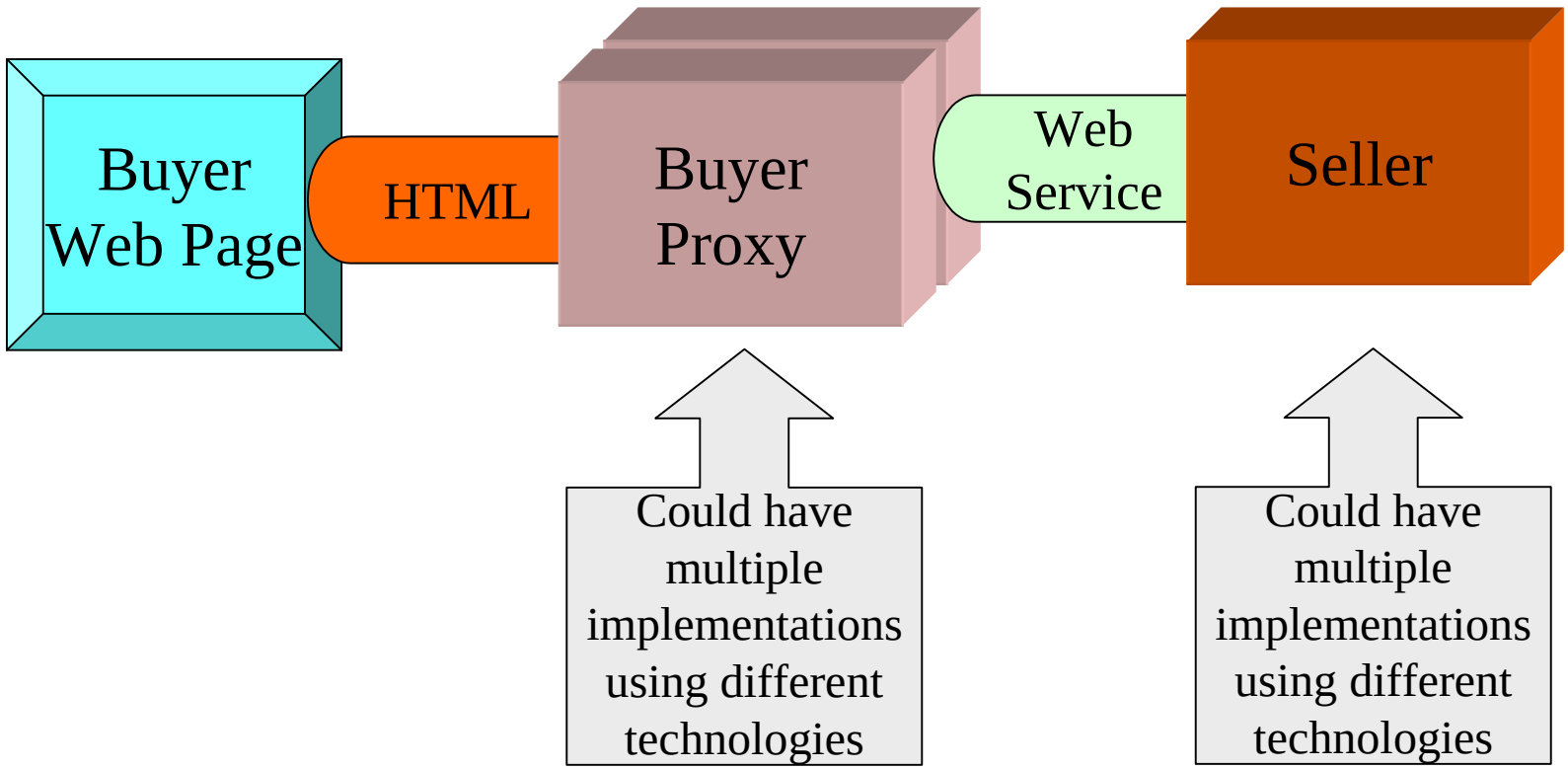
Technology Independence



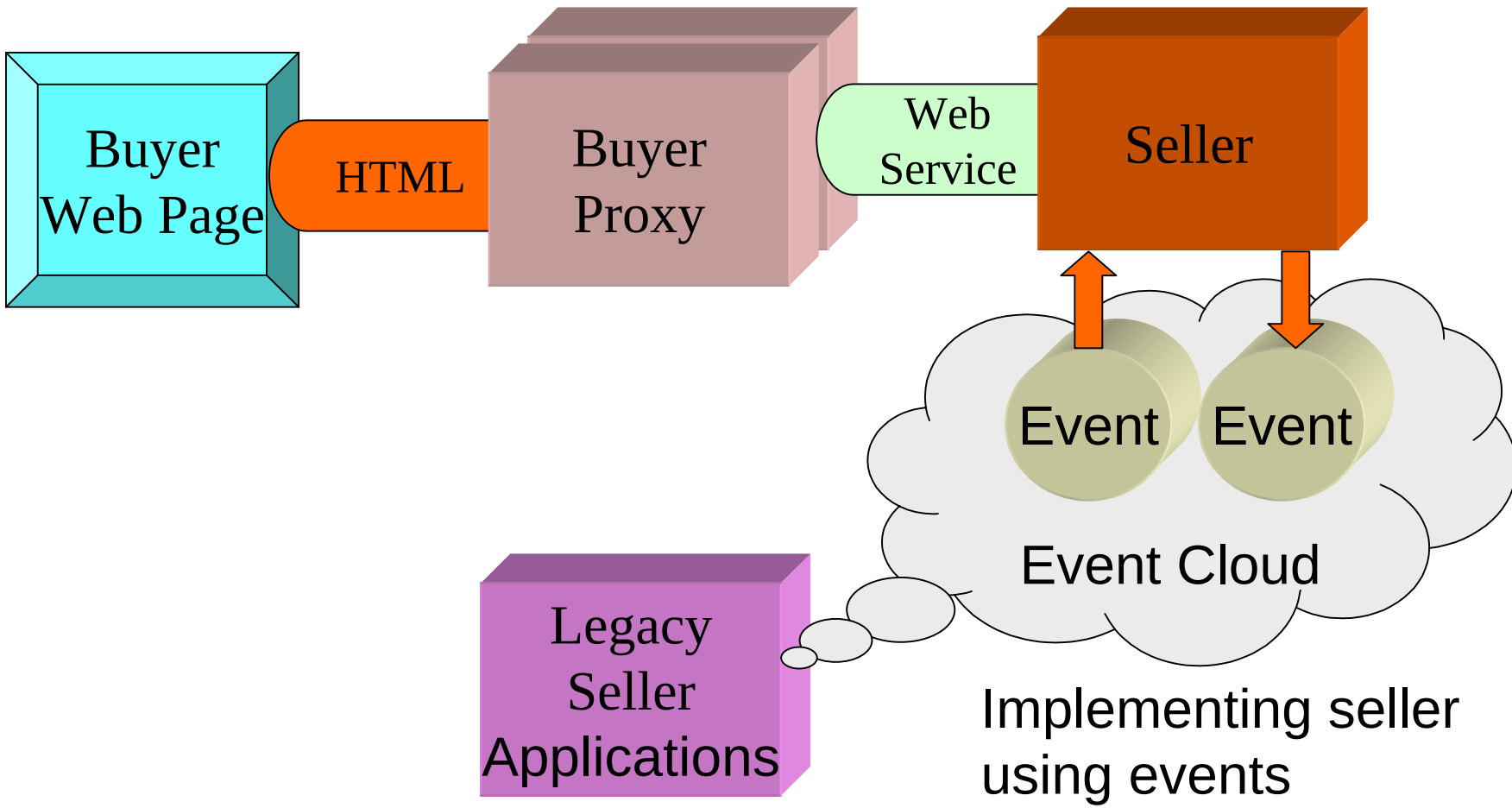
Typical Requirement



Multi-tier implementation



Multi-tier implementation



Implementing seller using events

Collaborations and Roles

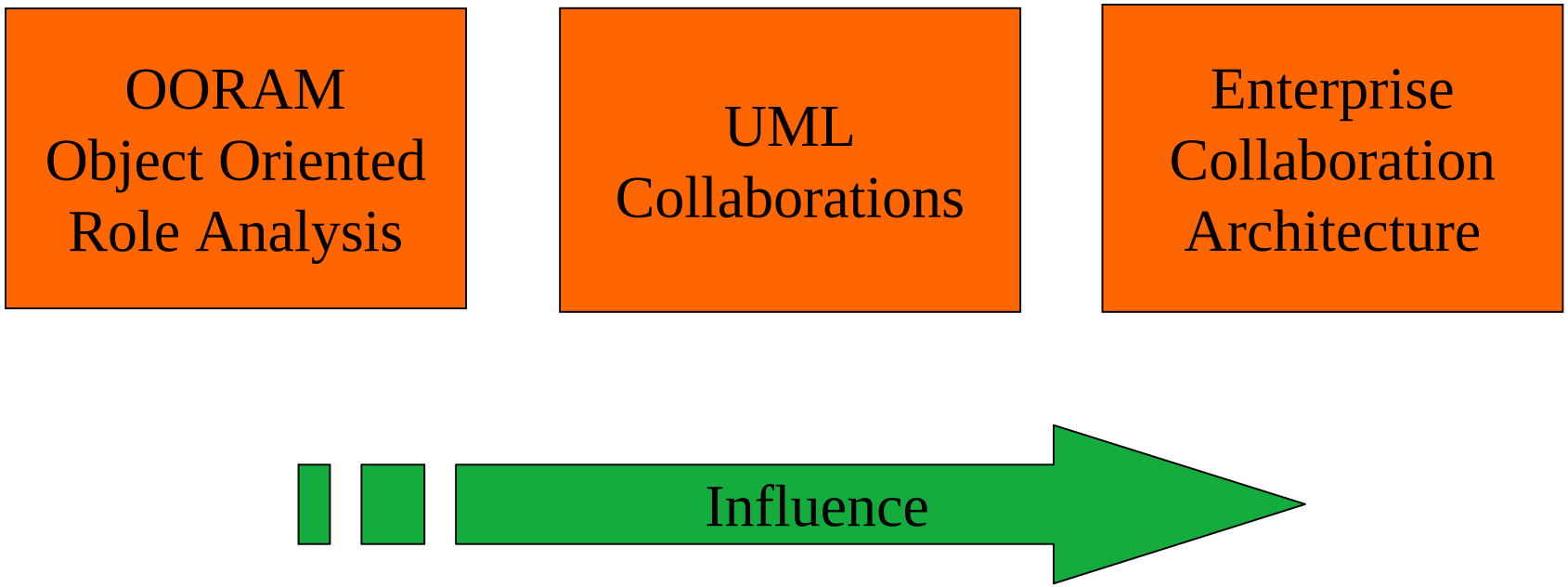
Conceptual Foundation

Portions copied with permission of Trygve Reenskaug - Taskon

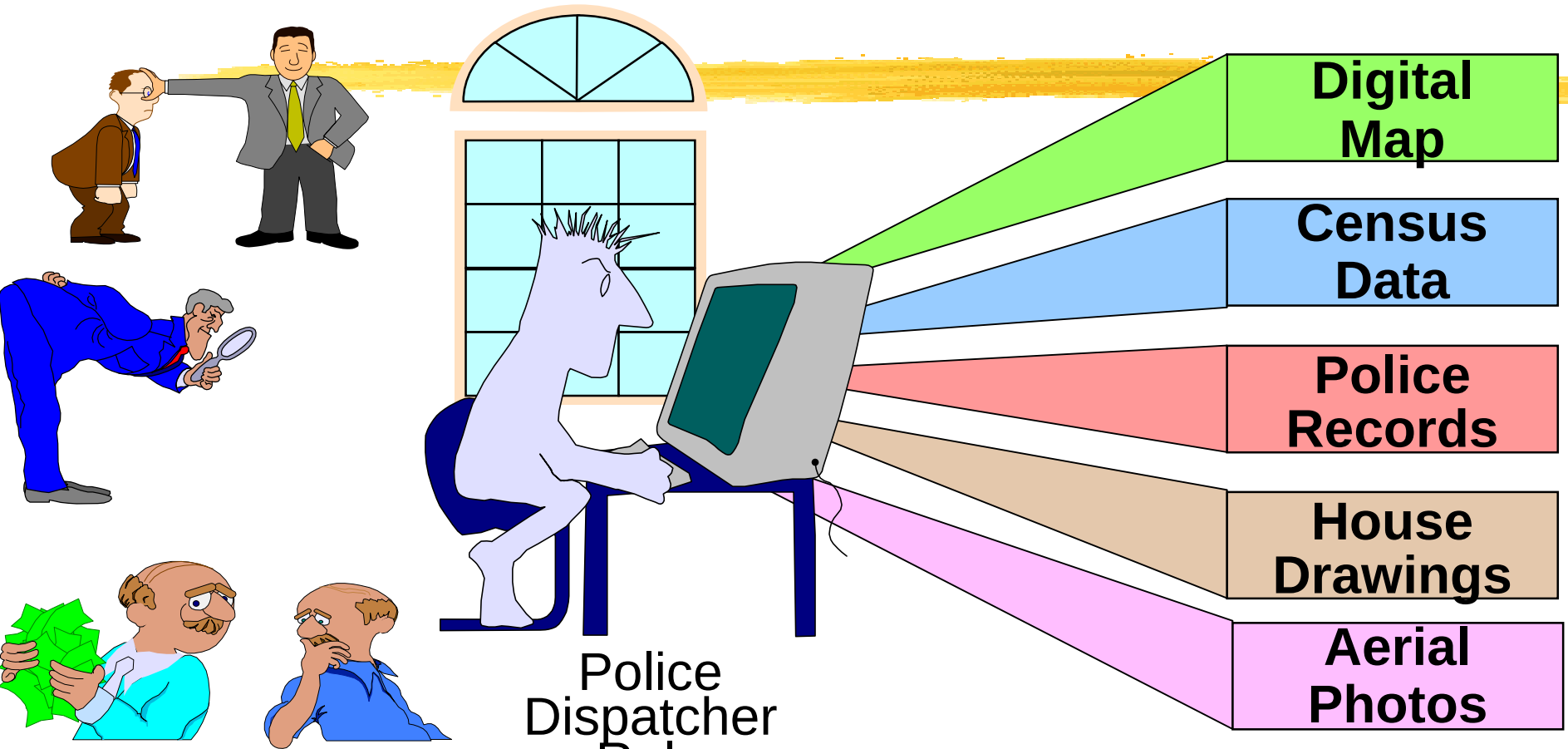
OORAM

(http://www.ifi.uio.no/~trygver)

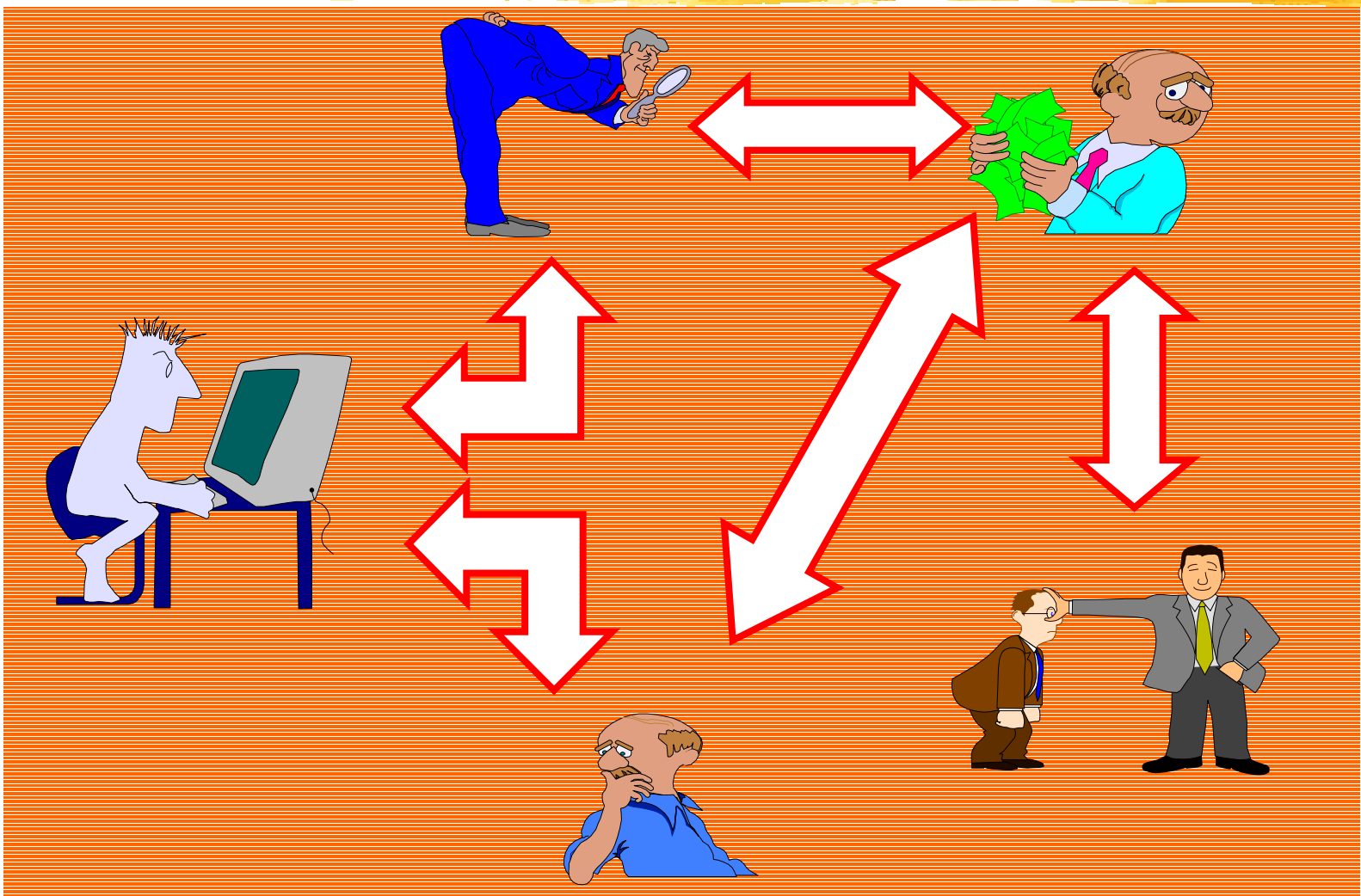
History



The Connected Enterprise
Content and Communication



Multiple roles in a collaboration



Travel Expense Example

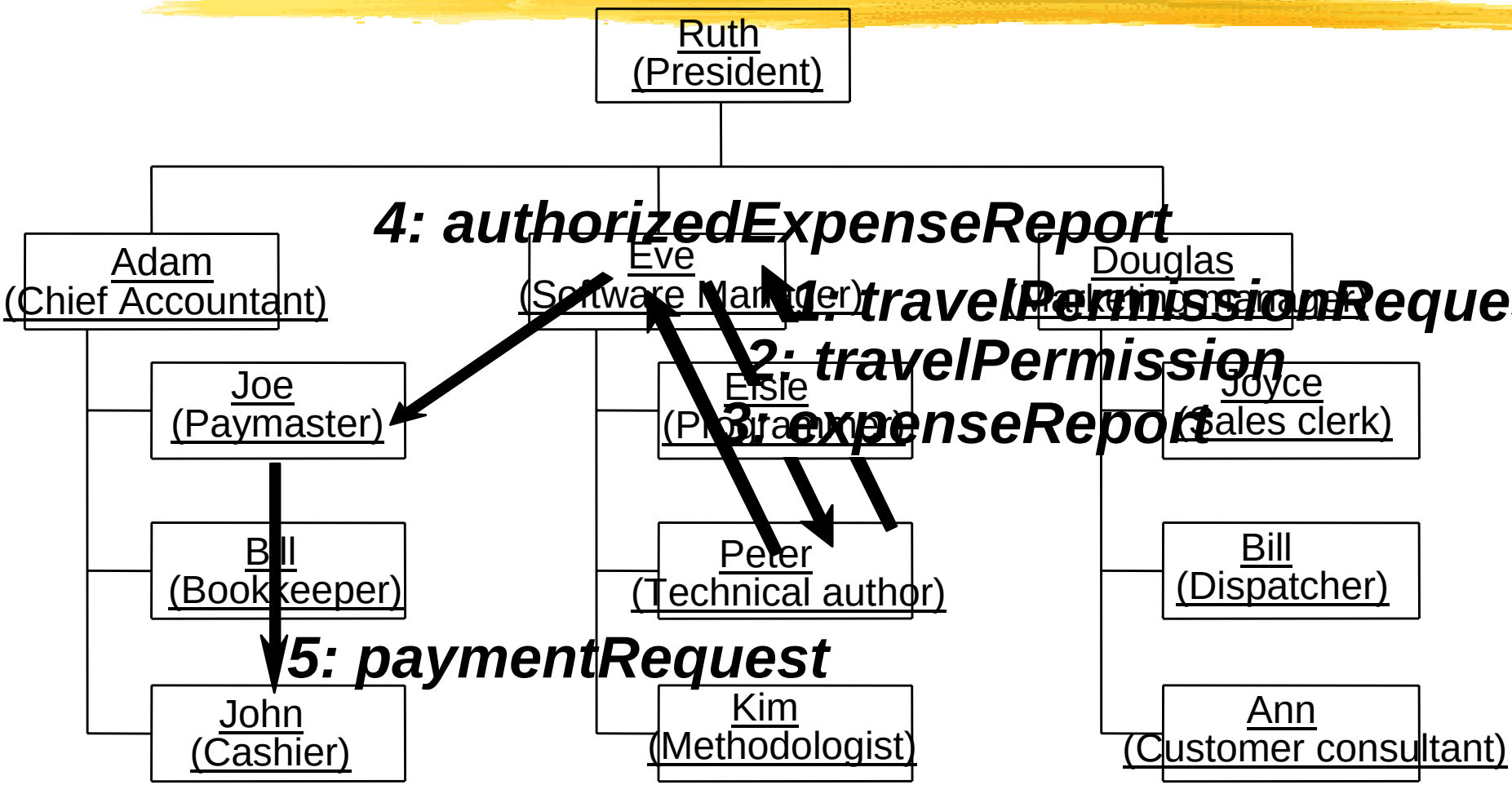
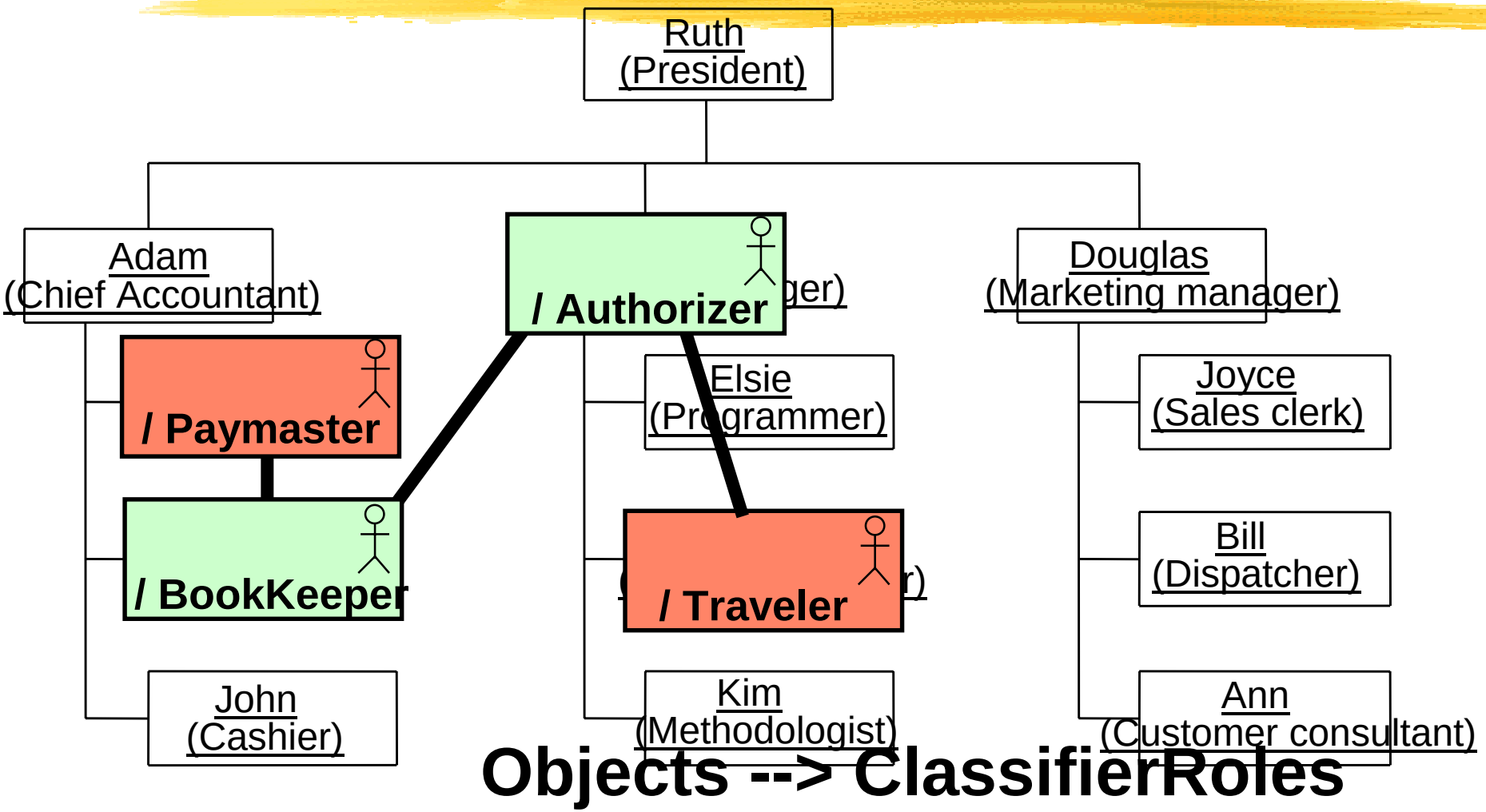
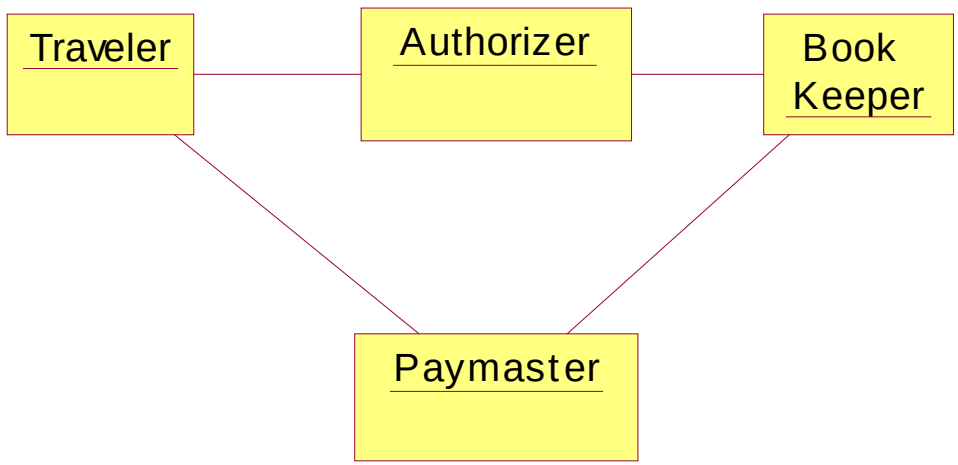


Diagram
Travel Expense Model

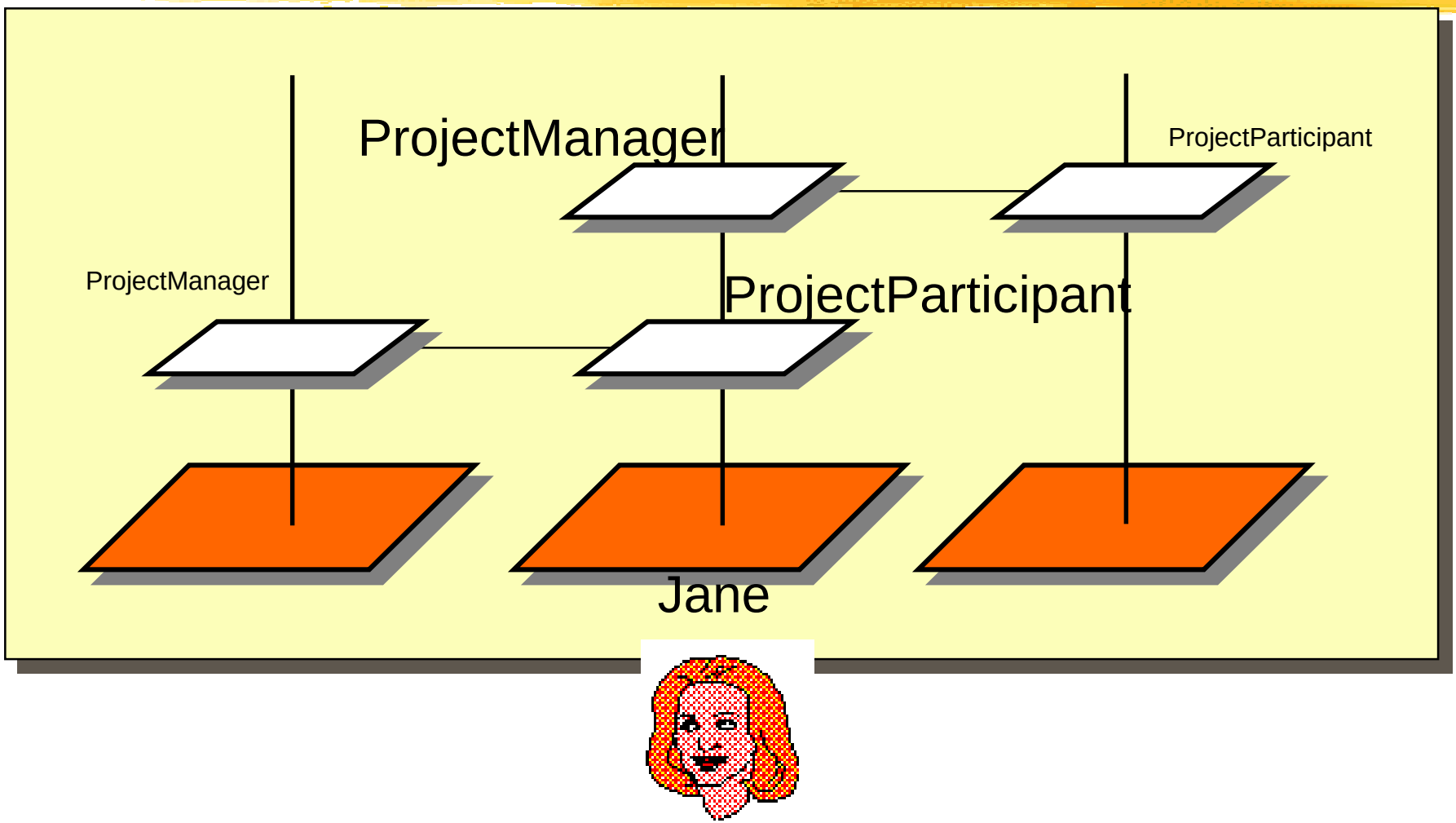


Objects --> ClassifierRoles

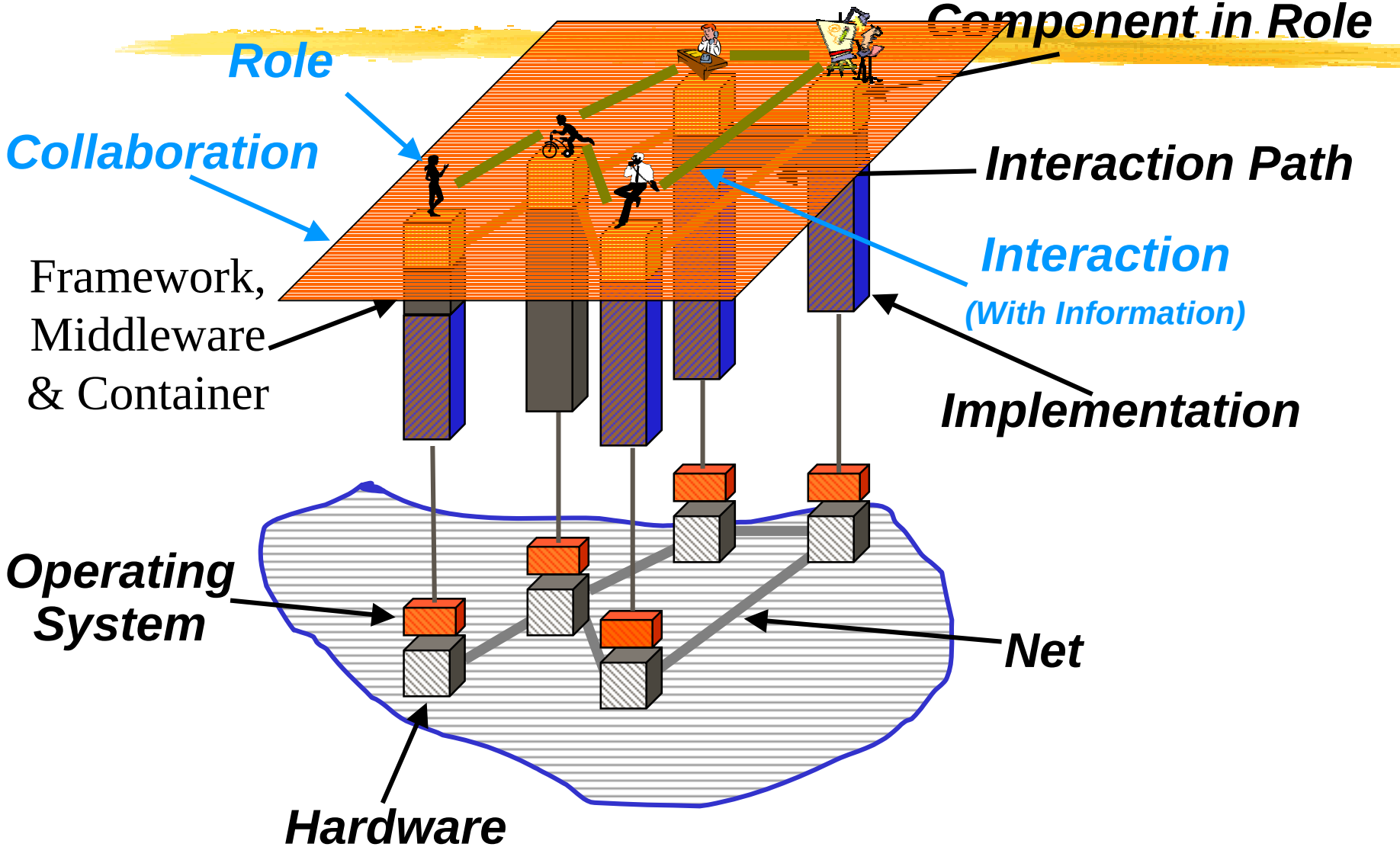
Collaboration Diagram



Synthesis - Components play several roles

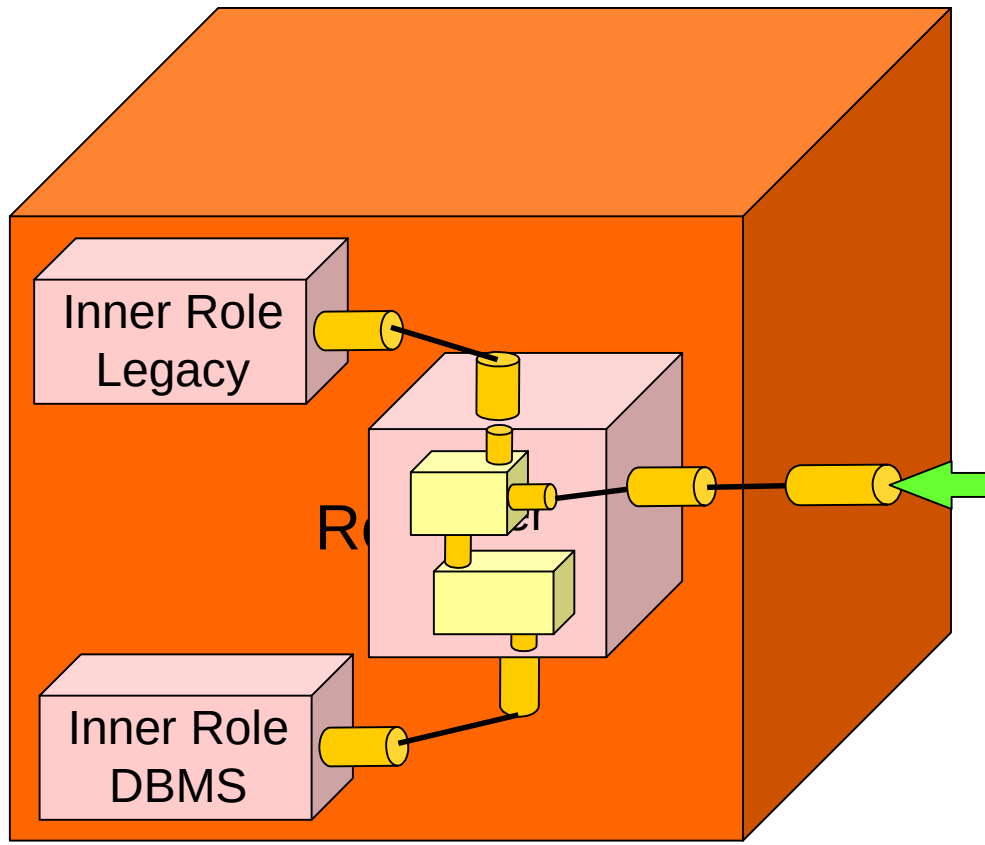


Roles to Systems



Drilling down – inside a role

- The open domain should make no assumptions about the "inside" of a role.
- Inside one role you *frequently* find more collaborating "parts" of the enterprise - the same model *may* be used
- Until you get to system inside a managed domain
  - Shared resources (DBMS)
  - Common Management
  - Frequently a legacy system



Discussion Point

- What are other ways of representing business semantics?
- What role do these play?
  - Business Processes
  - Information Models (DBMS)
  - Events
  - Workflow
  - Internet
  - Ontologies
- What is used internally?

Standards for Global Internet Computing

EDOC

XML

WSDL

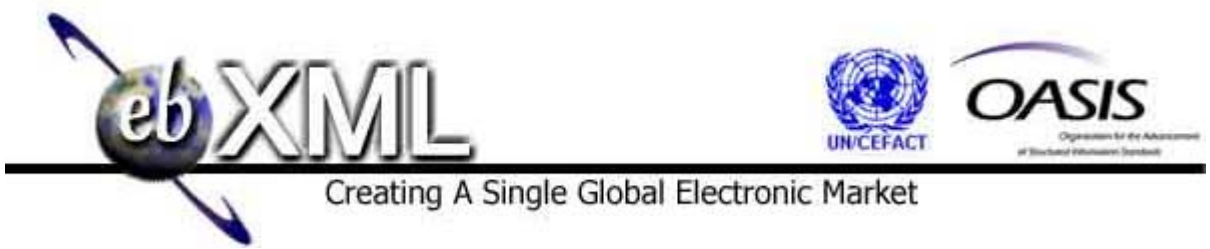
SOAP

XML-Schema

.NET

BPML

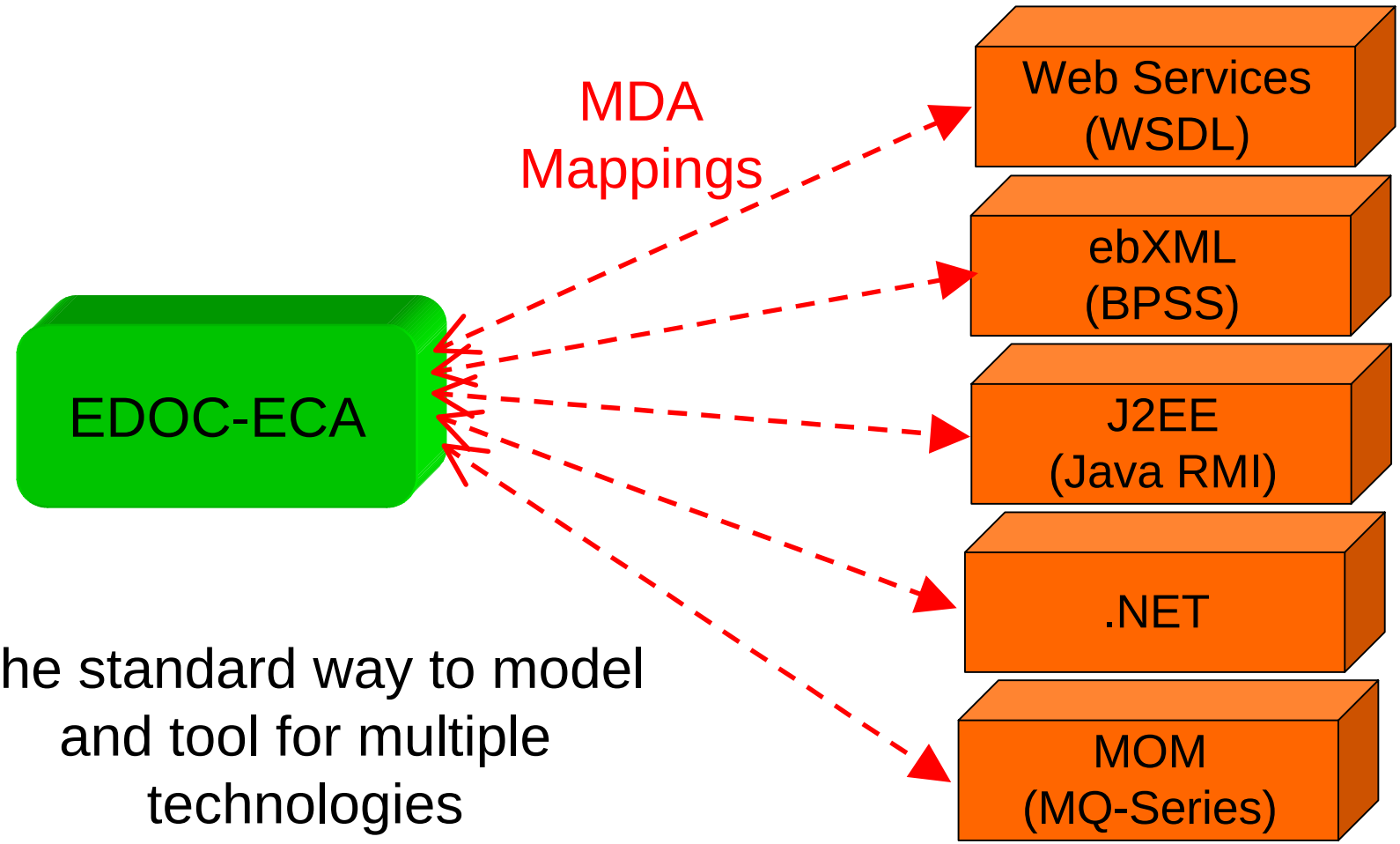
XLANG



XML Standards

- XML Schema & DTD
  - Description and packaging of data
- WSDL
  - Specification a services, operations and flows available via that service
- Soap
  - Basic messaging and packaging
  - Extensions for Soap-RPC with WSDL
  - May be extended to support collaborative messaging

ECA as the normal form



The standard way to model and tool for multiple technologies

Summary of points thus far

- We must enable the emerging Internet Computing Model
  - Loosely coupled roles exchanging documents based on a contract of collaboration
- Web need interoperability at two levels
  - Messaging for the data
  - Metadata for the contract of collaboration, stored in repositories
- This model of collaborating roles is recursive, extending into the enterprise, into managed domains and into applications
  - Inside the enterprise we want to include resources entities, business events and business processes
- Supporting the open domain has some required parts and can be augmented with a “treasure chest” of tools and infrastructure
- Between EDOC & ebXML we are covering B2B and intra enterprise

EDOC Component Collaboration Architecture

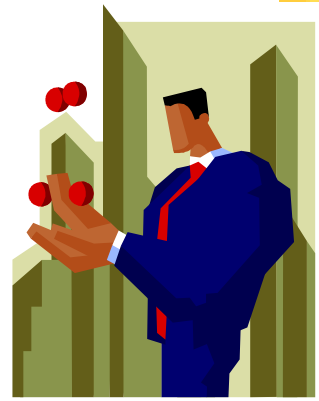
The model of collaborative work

The Marketplace Example

Mechanics Are Us
Buyer

Order

Conformation



Acme Industries
Seller

Process
Complete

Status

Ship Req

Shipped

Delivered

Physical
Delivery

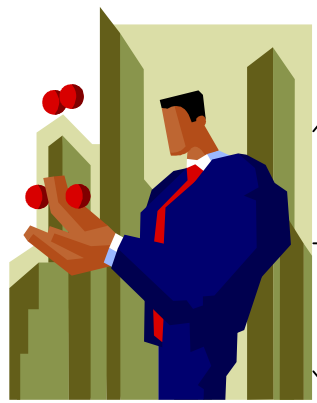
GetItThere Freight
Shipper

The Seller's Detail

Order

Conformation

Shipped



Order Processing

Shipping

Event

Receivables

Ship Req

Shipped

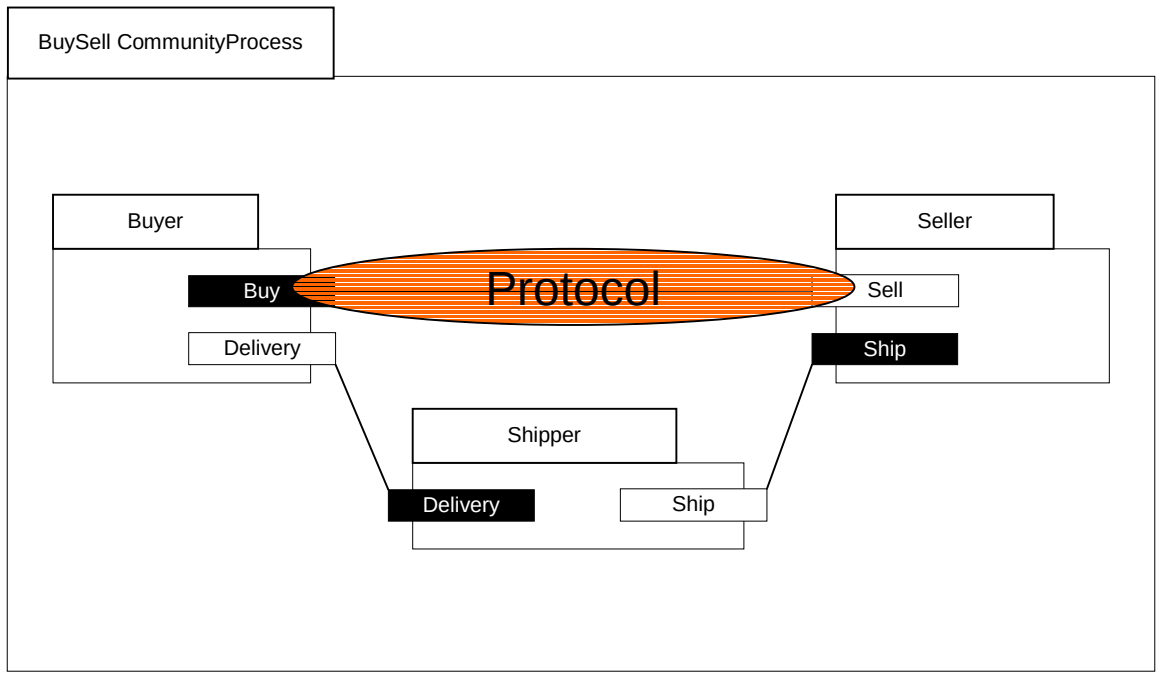
Delivered

Parts of a CCA Specification

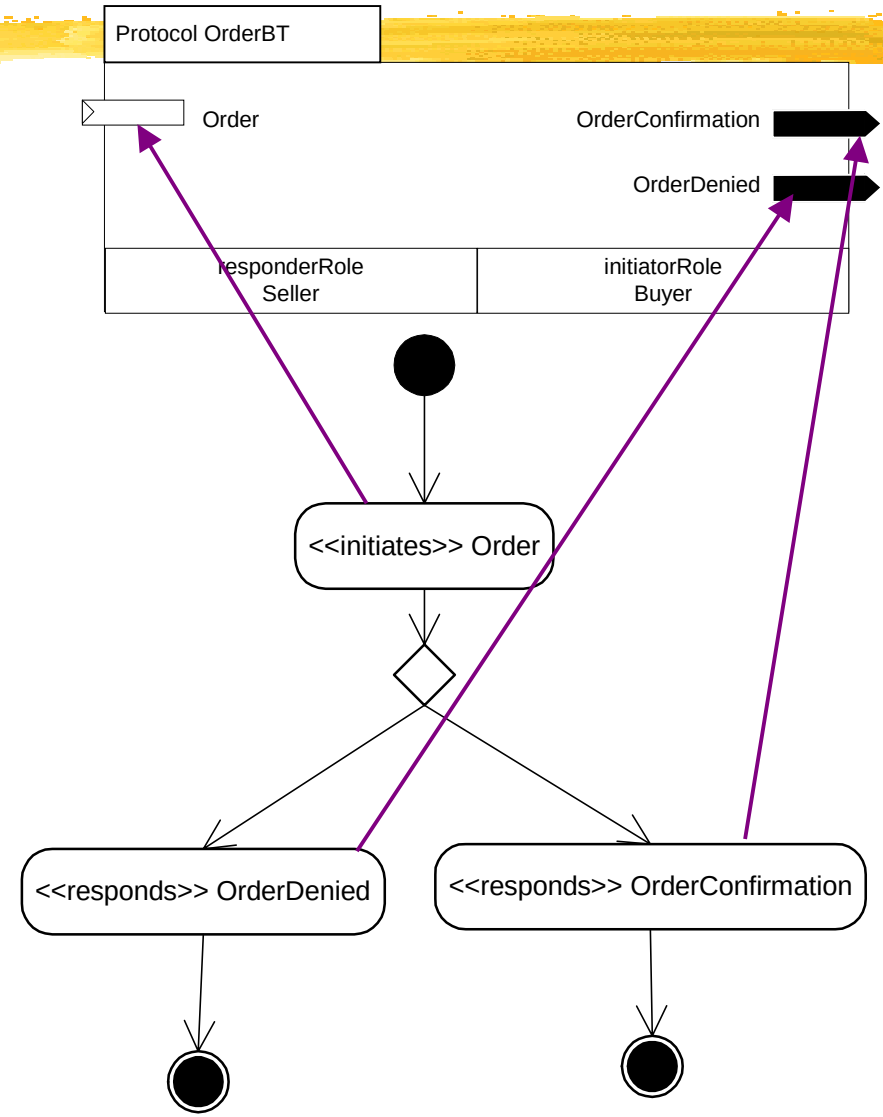
- Structure of process components and protocols
  - Process components, ports, protocols and documents
    - Class Diagram or CCA Notation
- Composition of process components
  - How components are used to specify components
    - Collaboration diagram or CCA Notation
- Choreography
  - Ordering of flows and protocols in and between process components
    - Activity Diagram

The Community Process

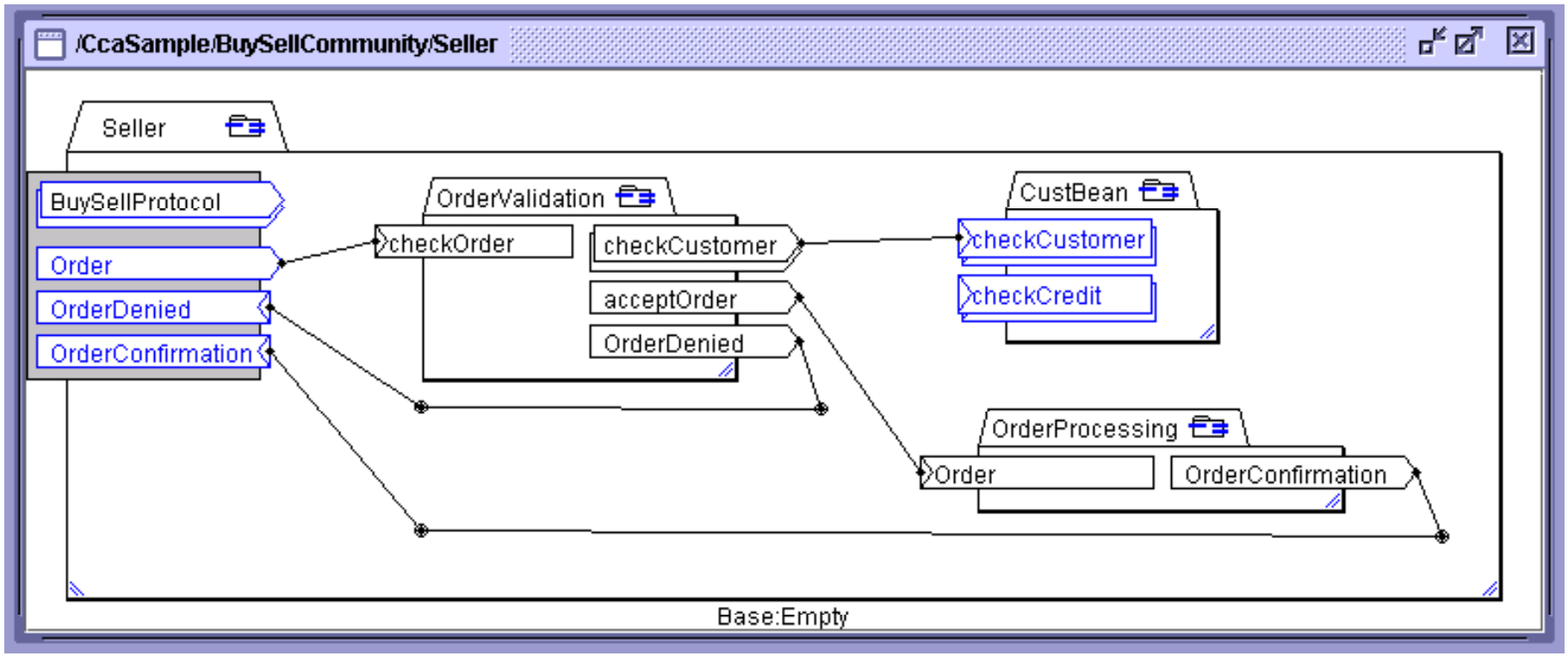
- Identify a “community process”, the roles and interactions in a collaboration



Protocols



Composition



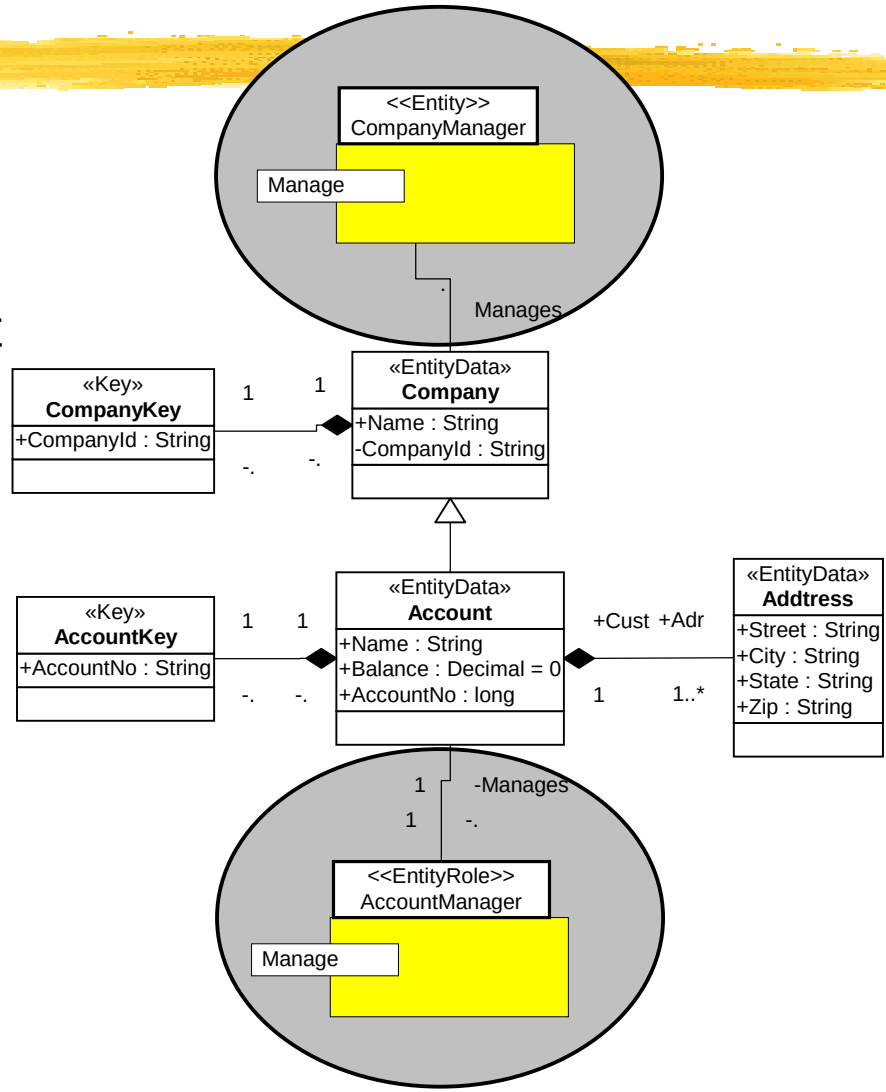
ECA Entity Profile

The model of things

Data Inside a “shared domain”

Adding Entities

- Entities are added to manage entity data
- Entity Roles are managers that provides a view of the same identity in another context
- The Entities have ports for managing and accessing the entities
- Non-entities which are owned by (aggregate into) an entity are managed by the entity

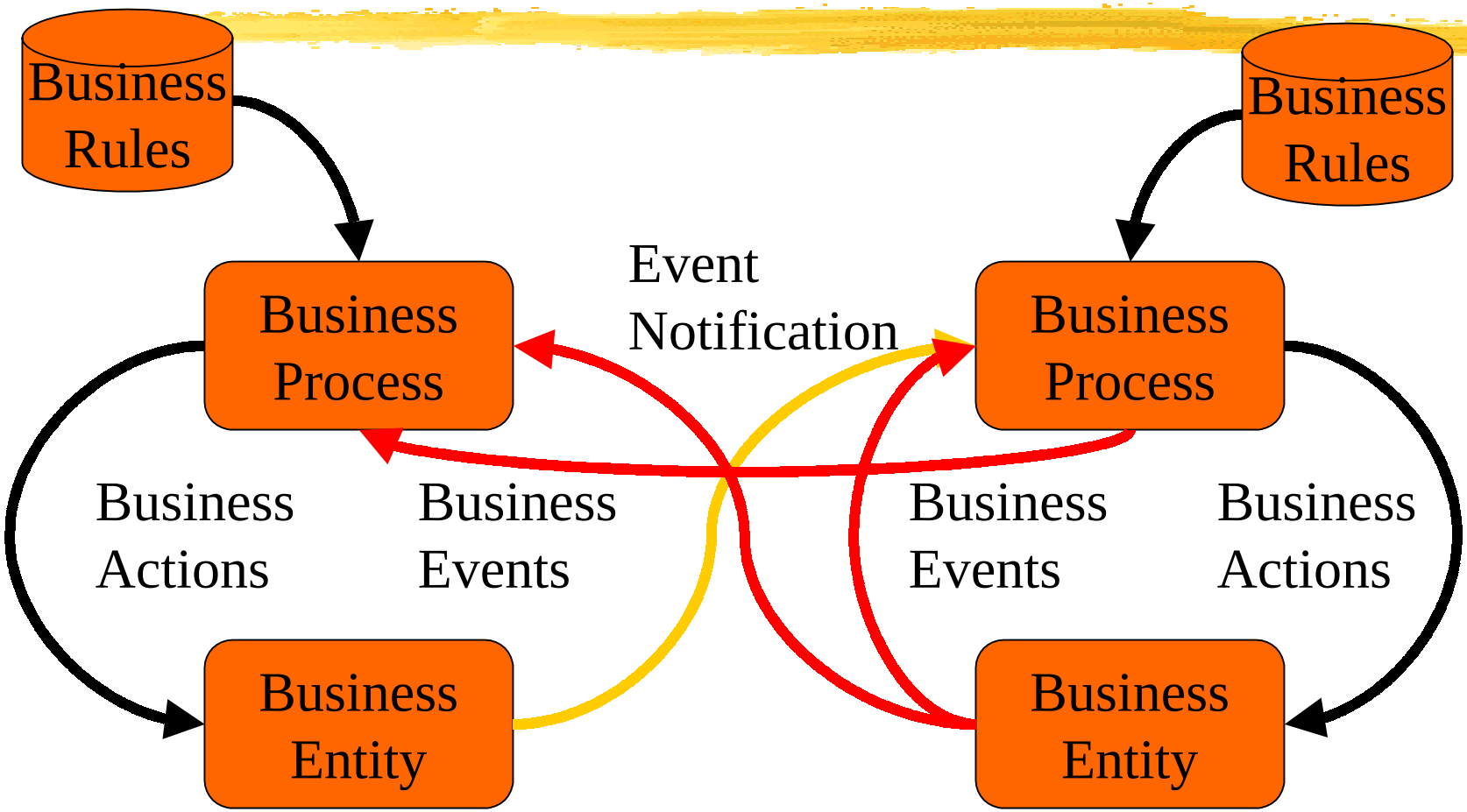


ECA Business Events

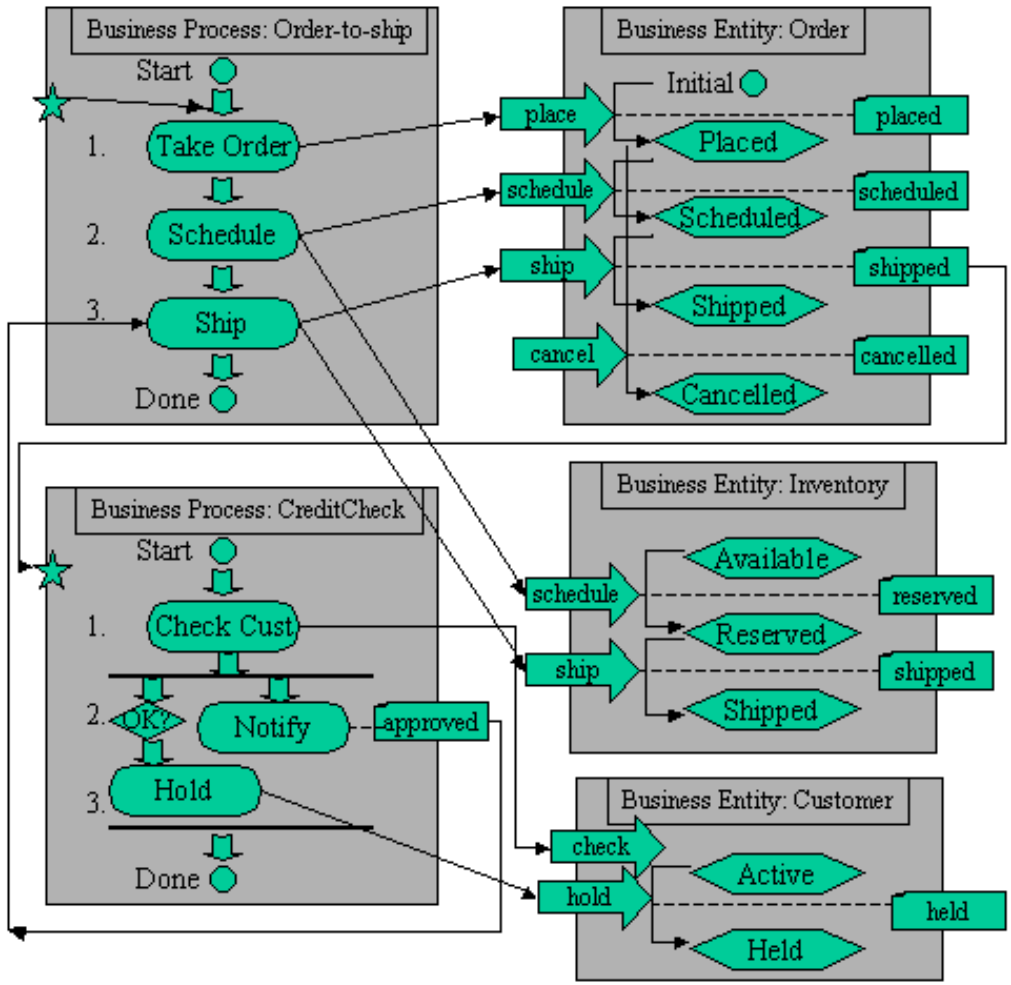
The model of when...

Loosely coupled integration within the enterprise and with “aligned” business partners

Event Based Business Processes



Event Example



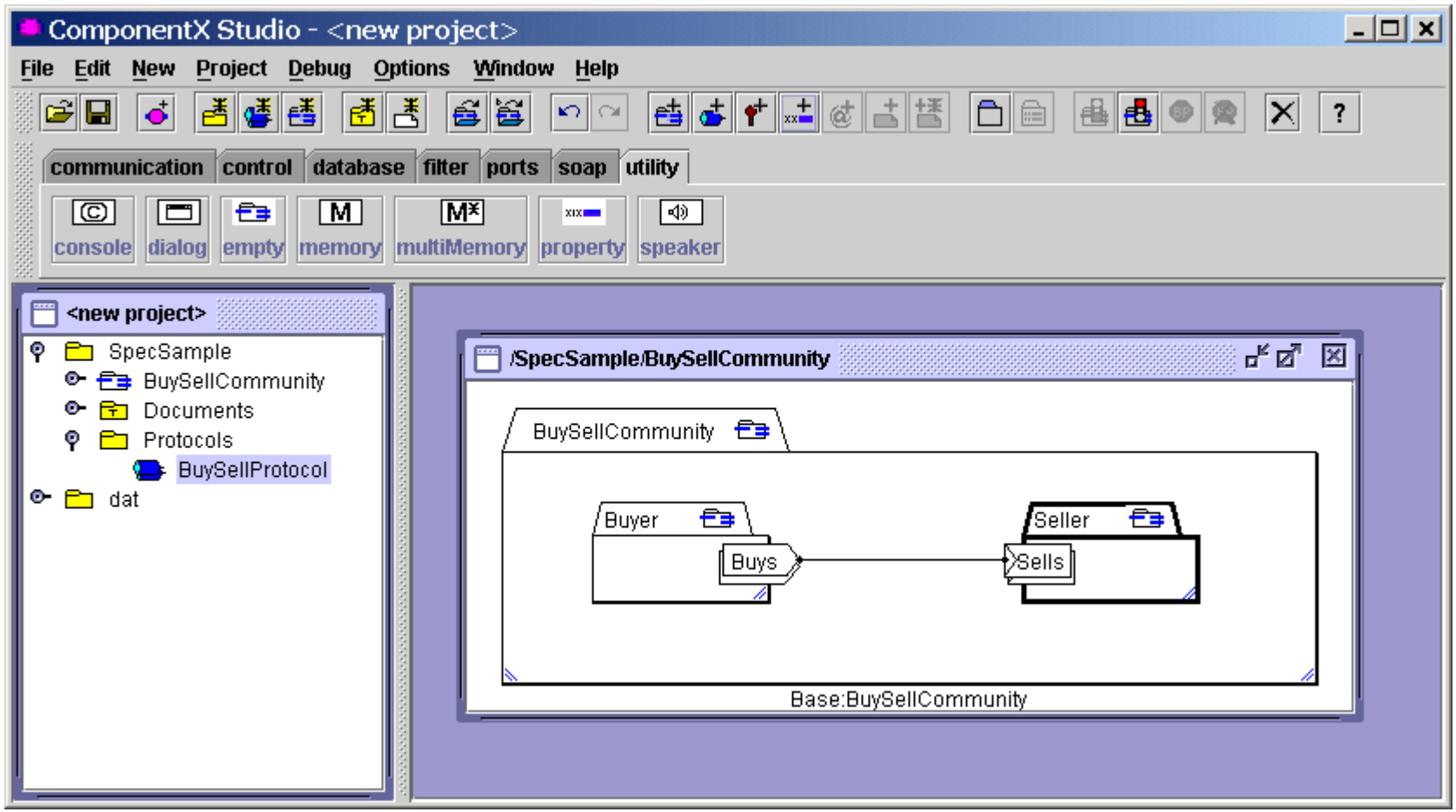
Events or Services

- Services - conversation between known parties
  - Good supply chain model
- Events - Actions based on notification of events
  - Parties do not know about each other!
  - More loosely coupled
  - Works best inside a managed domain (enterprise)

EDOC CCA Example

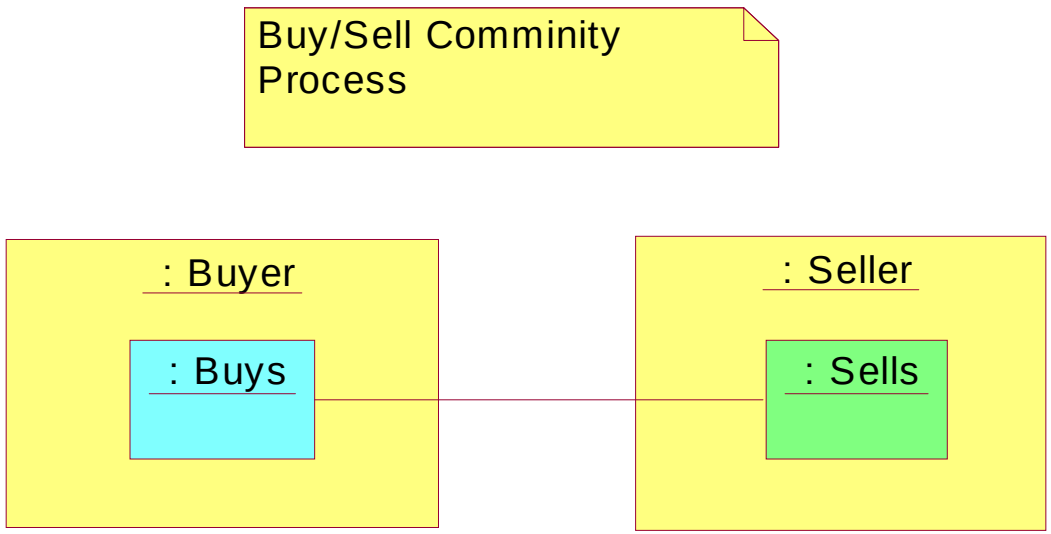
Buy-Sell Process

Community Process (CCA)



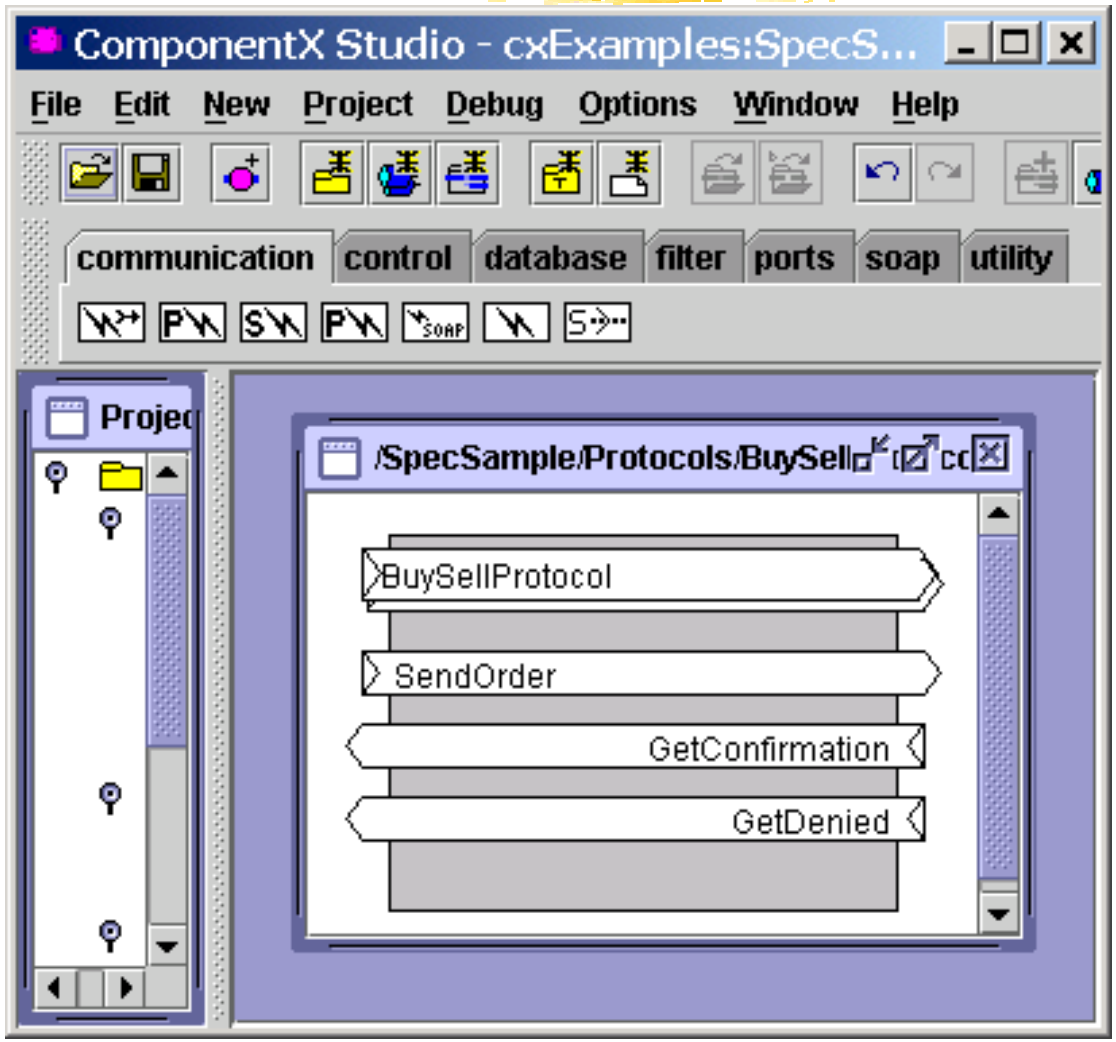
CCA Notation

Community Process

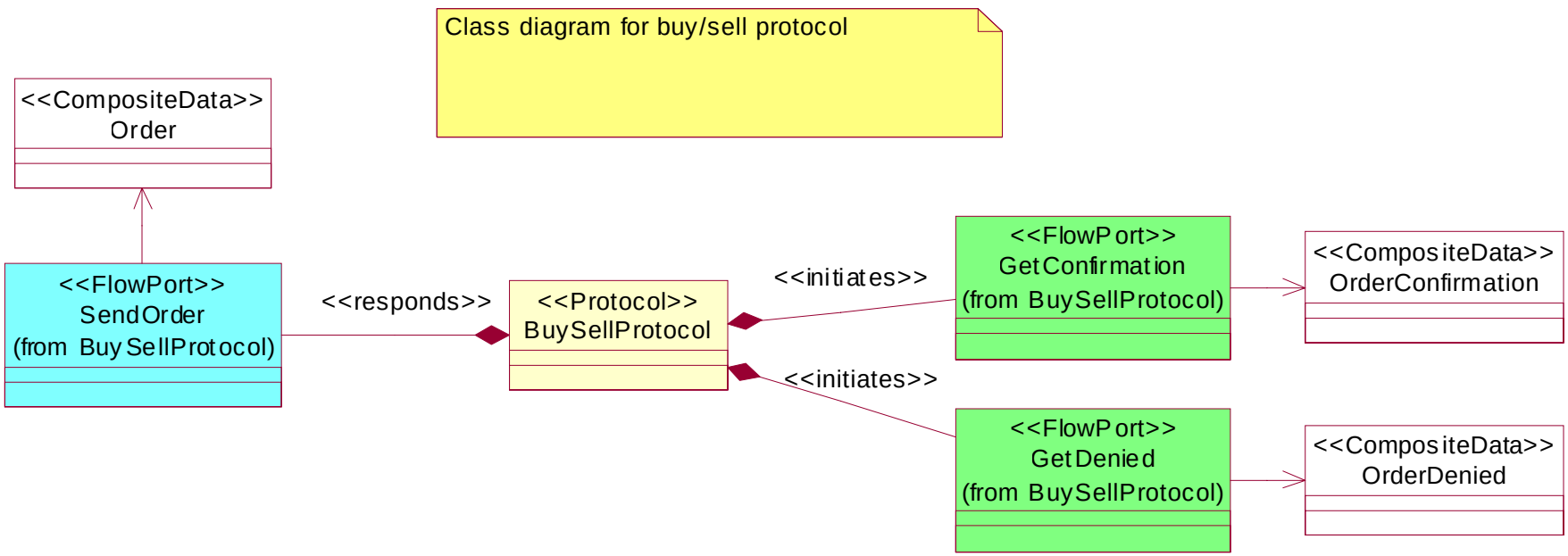


UML Collaboration Diagram

Protocol (CCA)

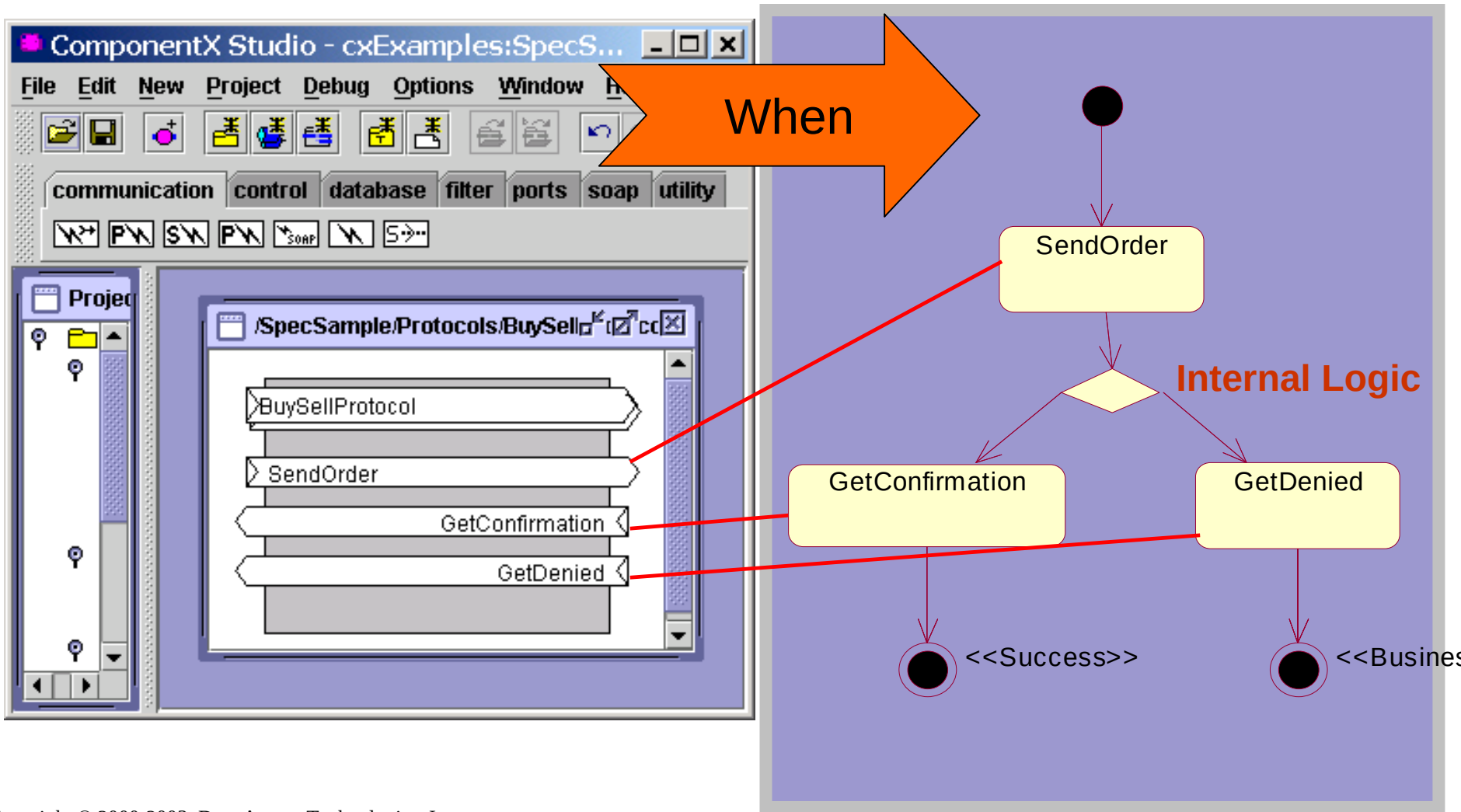


Protocol



UML Class Diagram

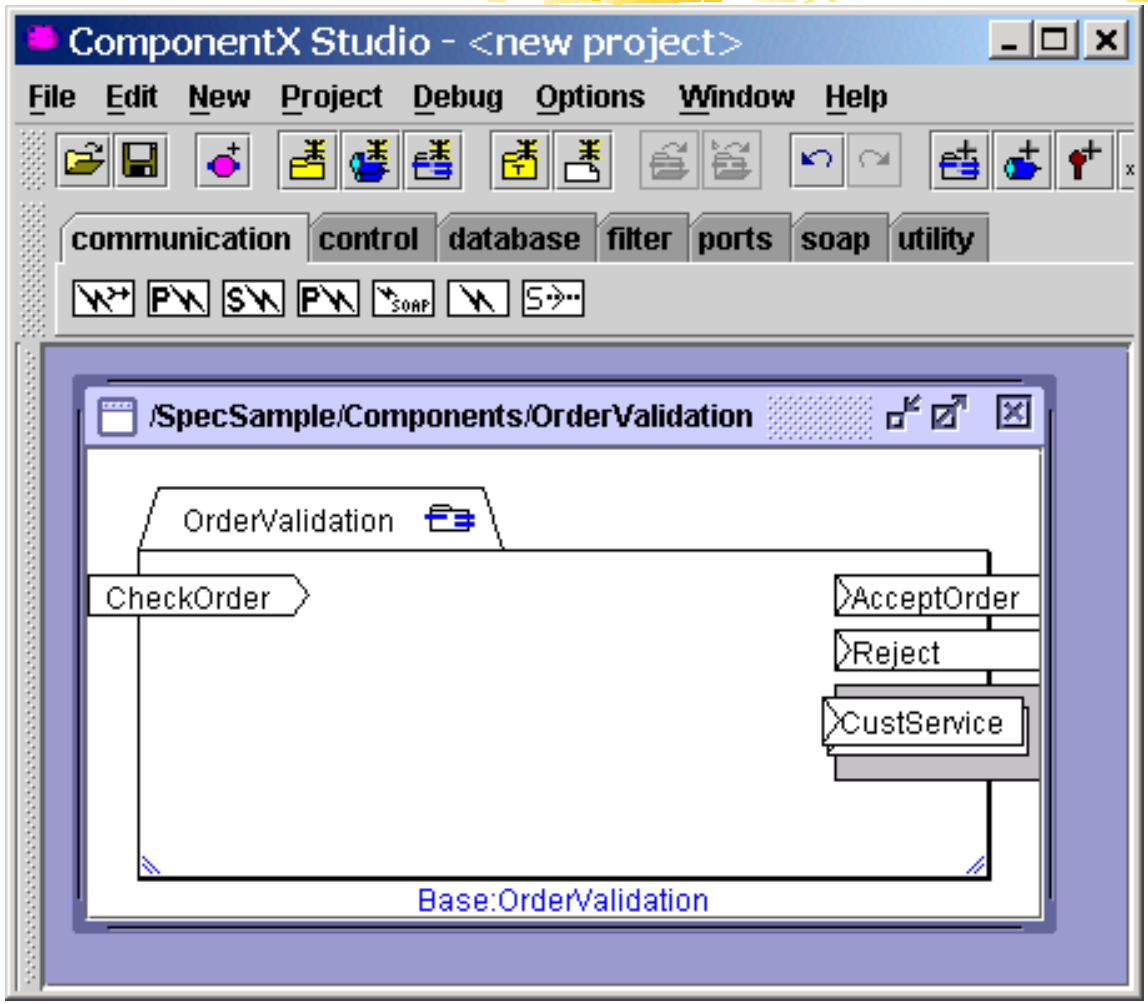
Protocol Choreography



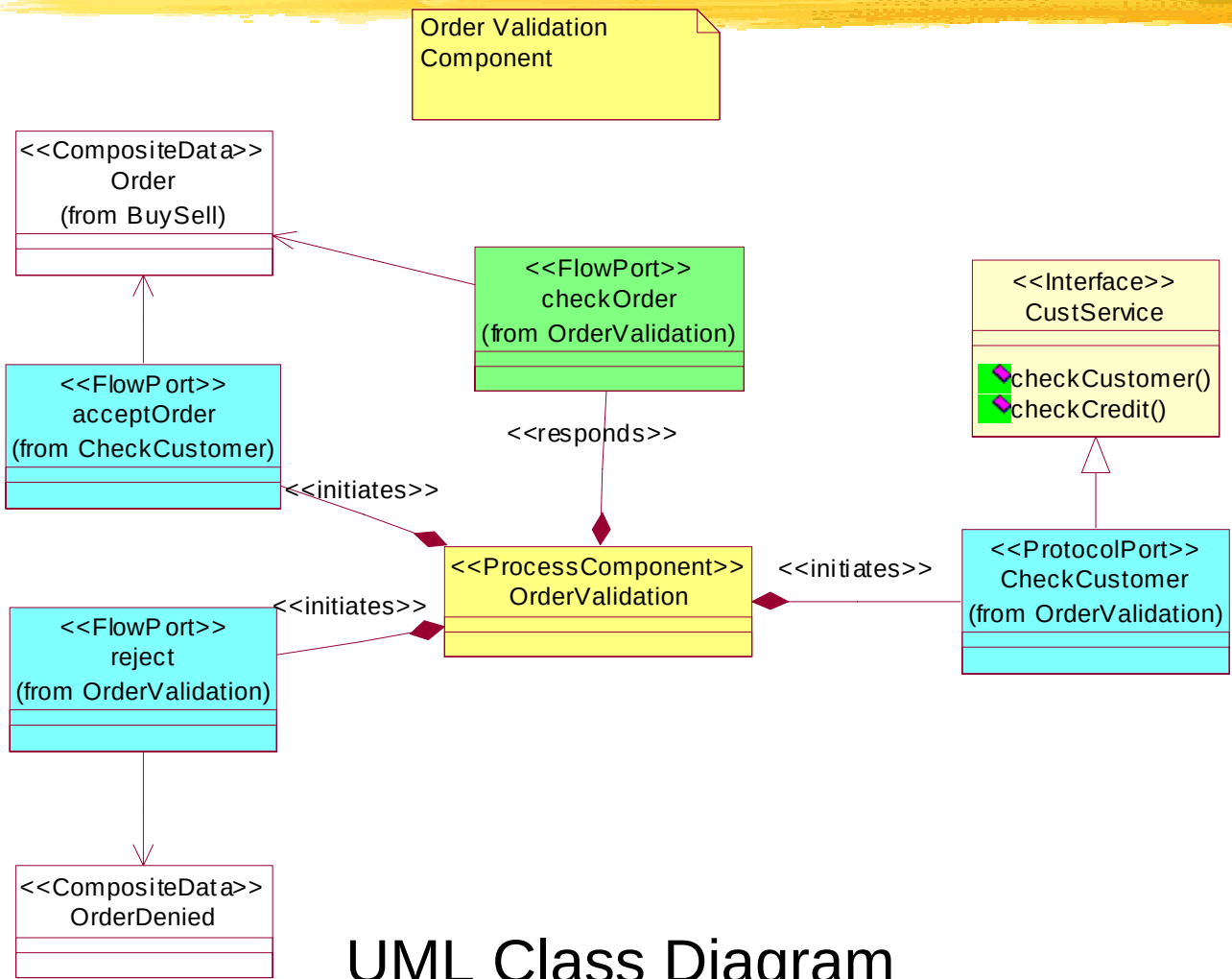
EDOC Exercise - Community

- Part one
  - Pick a collaboration from the first exercise
  - Model it with EDOC notation as a community process
- Identify an interaction
  - Specify the documents of the interaction
    - What are some of the major elements?
  - Specify any choreography

Validation Component (CCA)

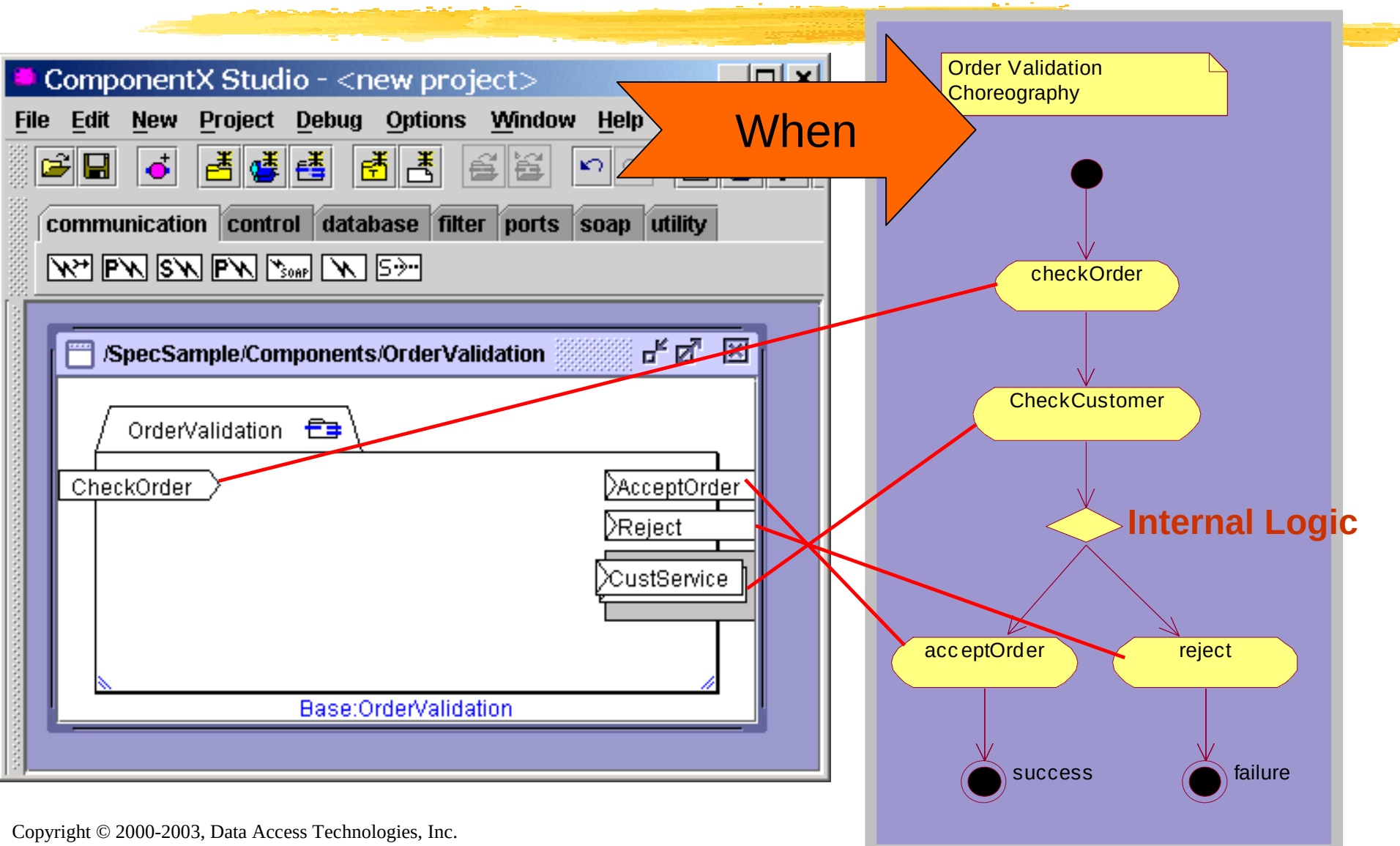


Validation Component

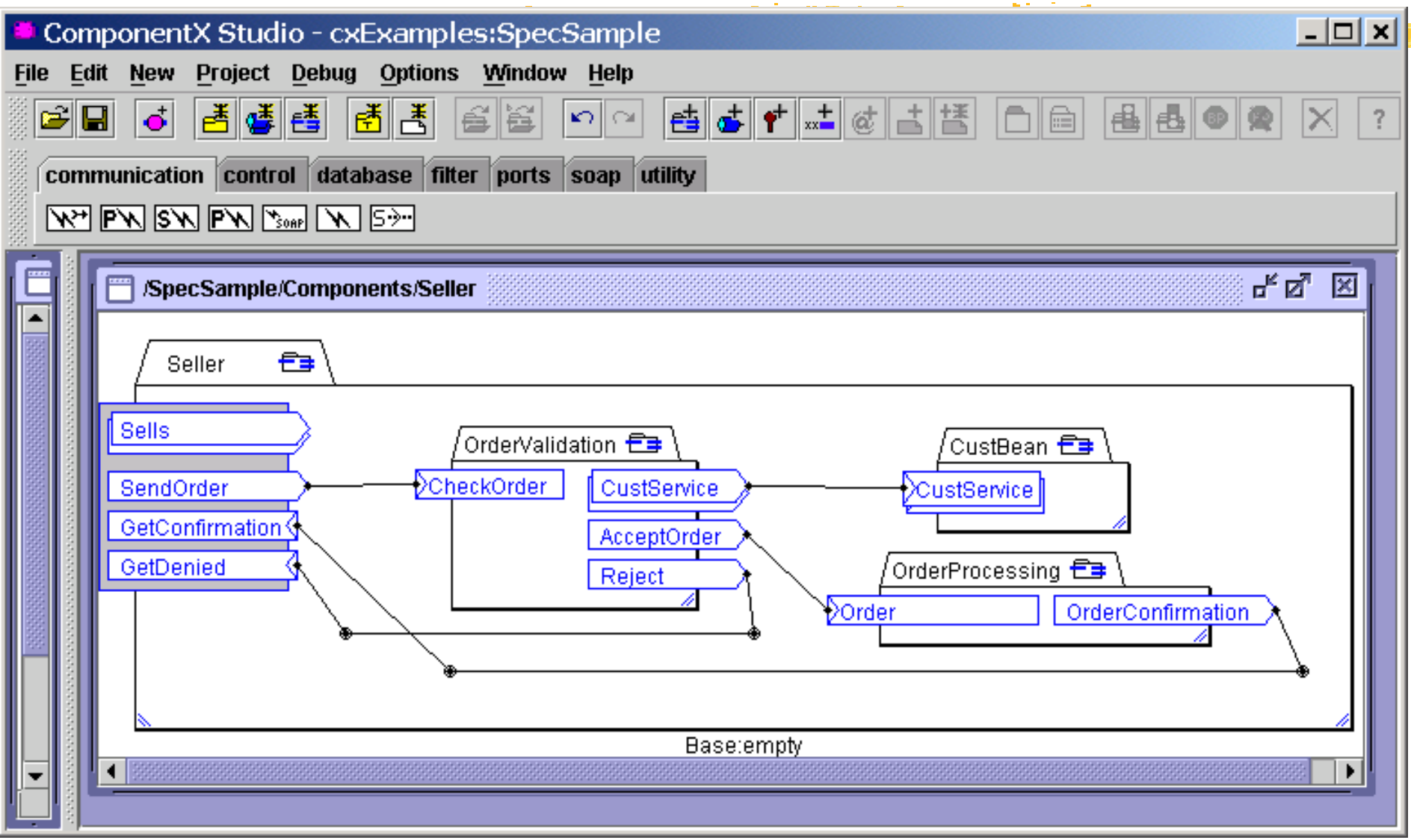


UML Class Diagram

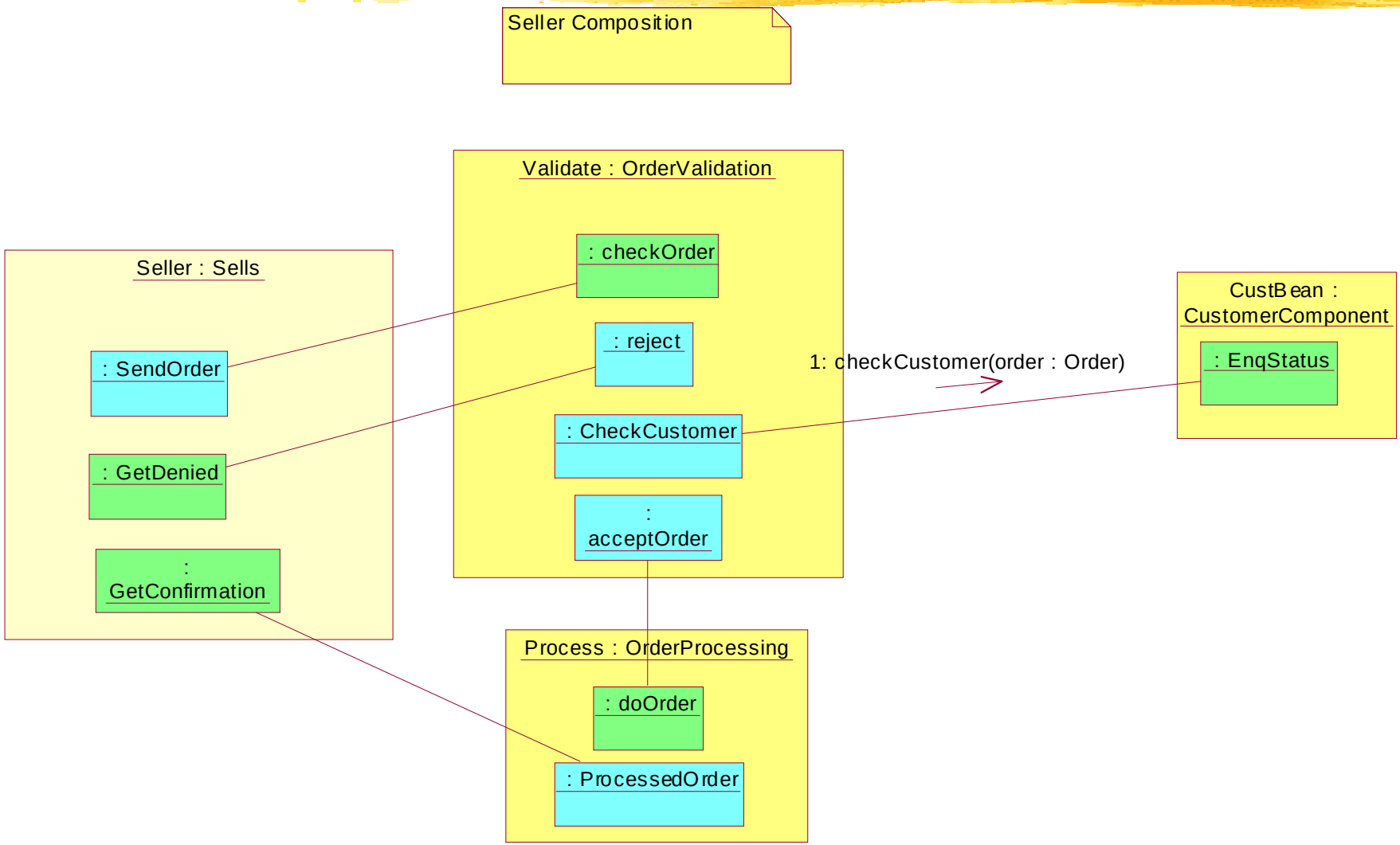
Choreography



Composition (CCA)



Composition (UML Collaboration)



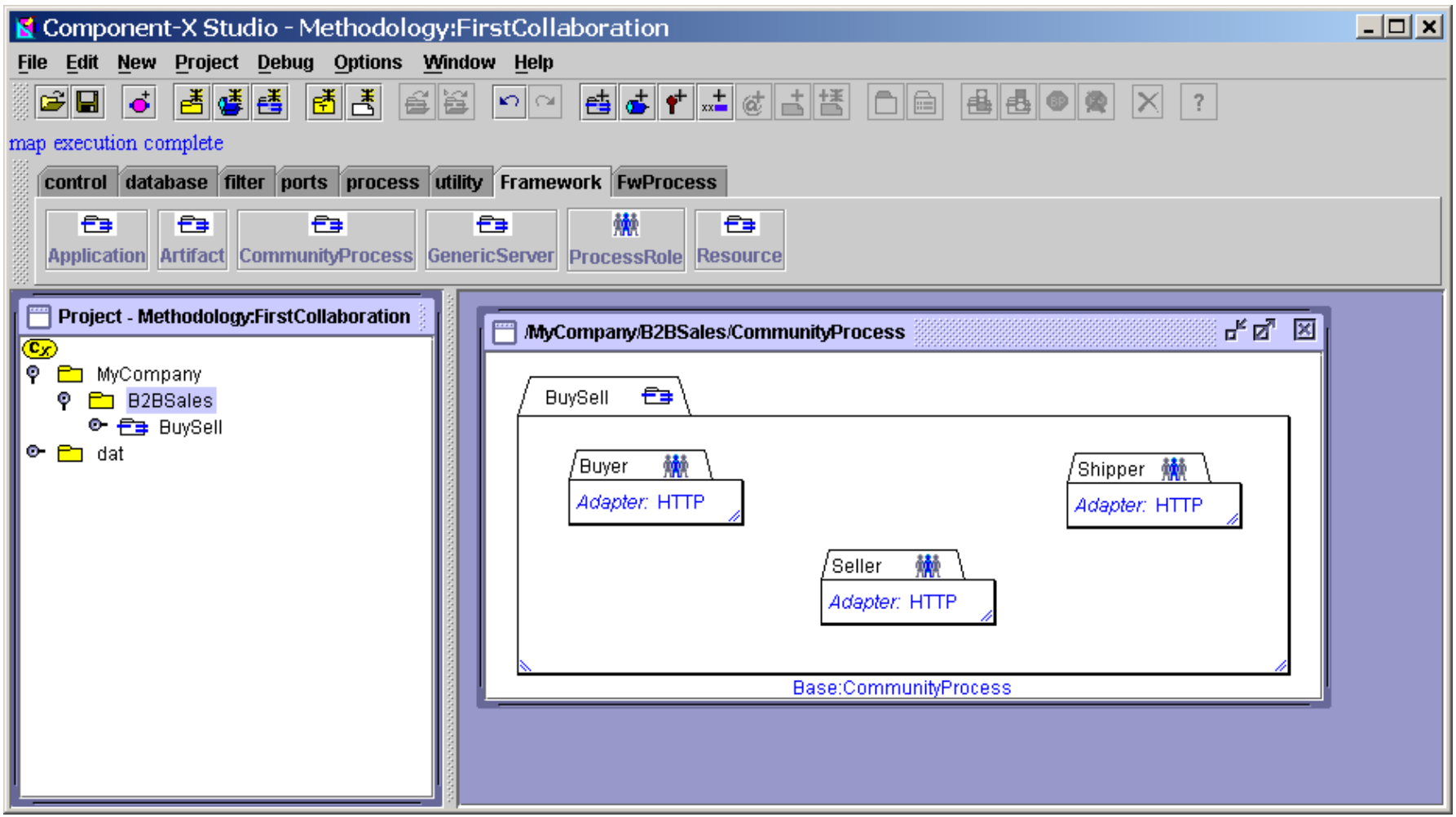
ECA Methodology

A simple methodology for creating collaborative bueiness processes

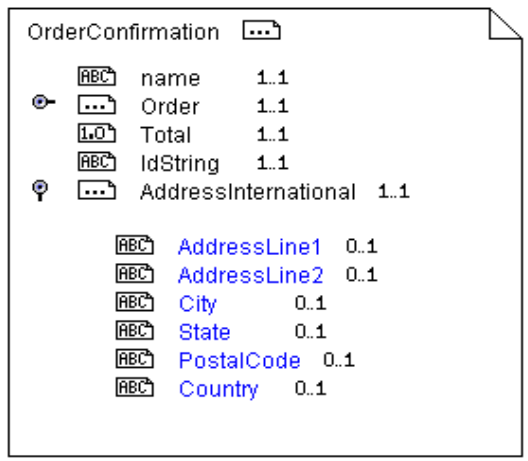
Basic Steps

- Identify roles and organize roles into collaborations
- Define collaboration documents
- Create basic business transactions
- Organize into protocols and events
- Use protocols to define ports on roles
- Drill-down into role detail
- Implement roles
- Configure implementations for deployment with technology specifics
- Deploy

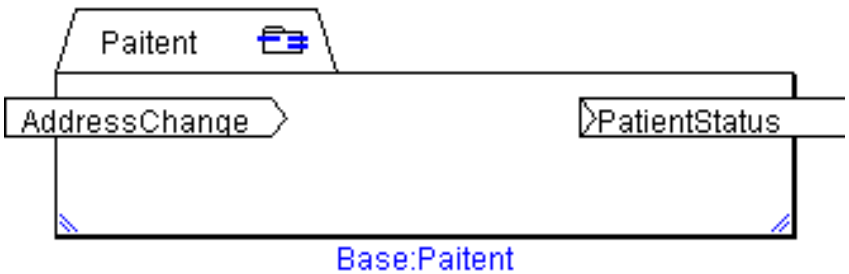
Identifying roles and collaborations

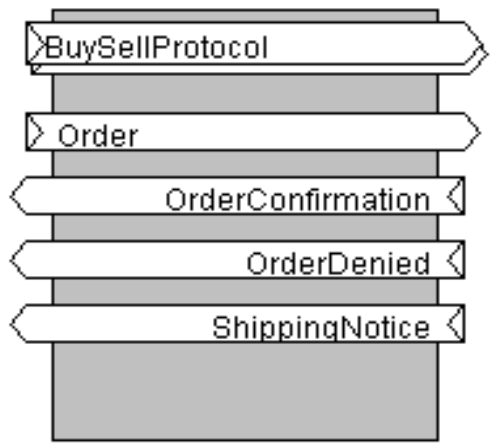


Identify Documents

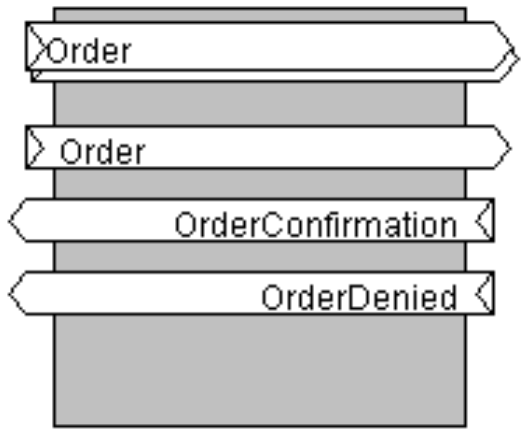


Distinguish protocols and events

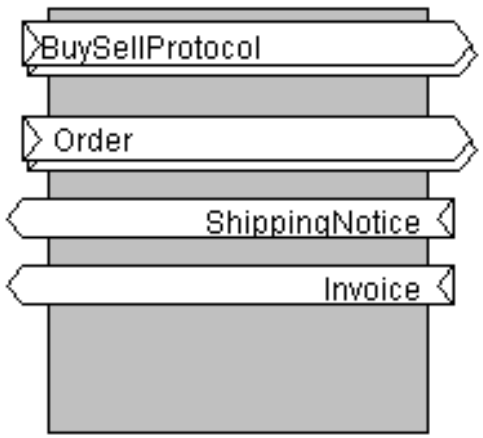


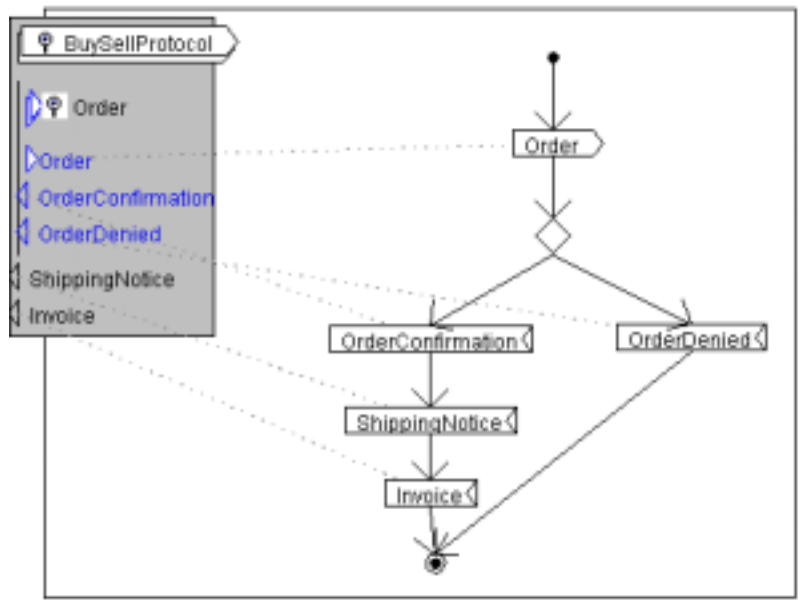


Create Business Transactions

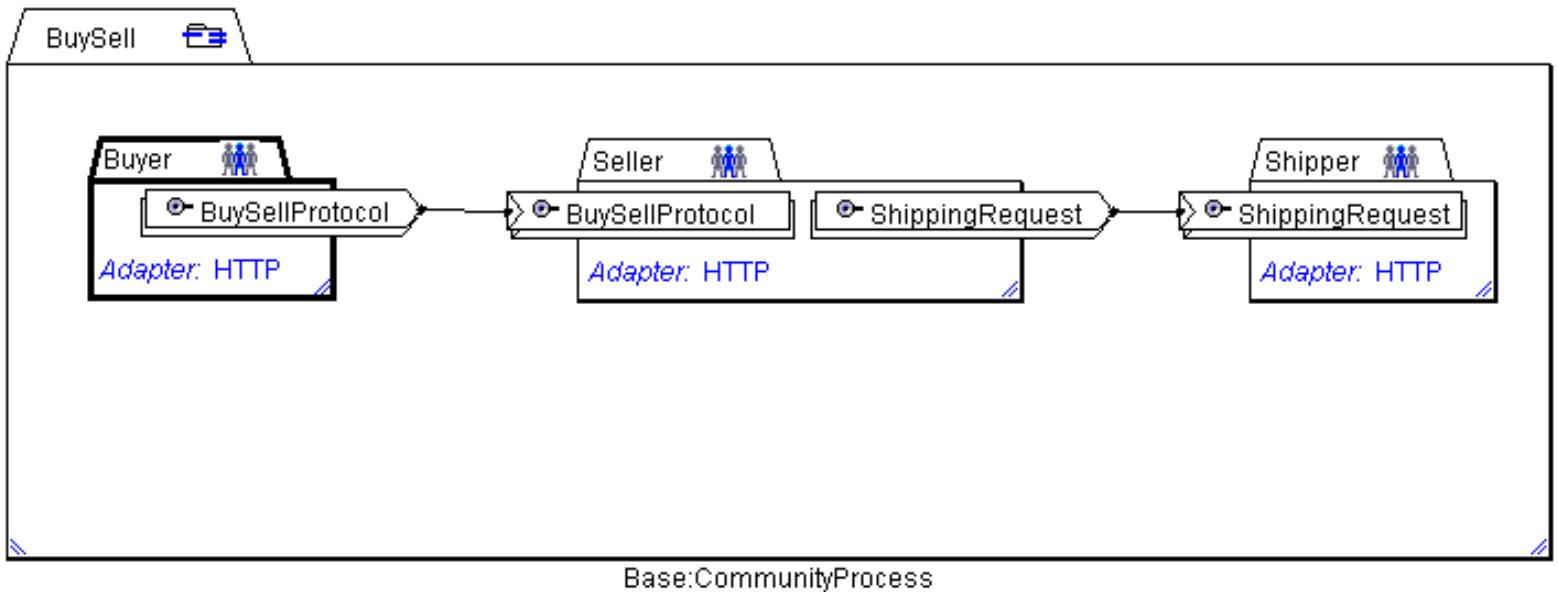


Organize into protocols

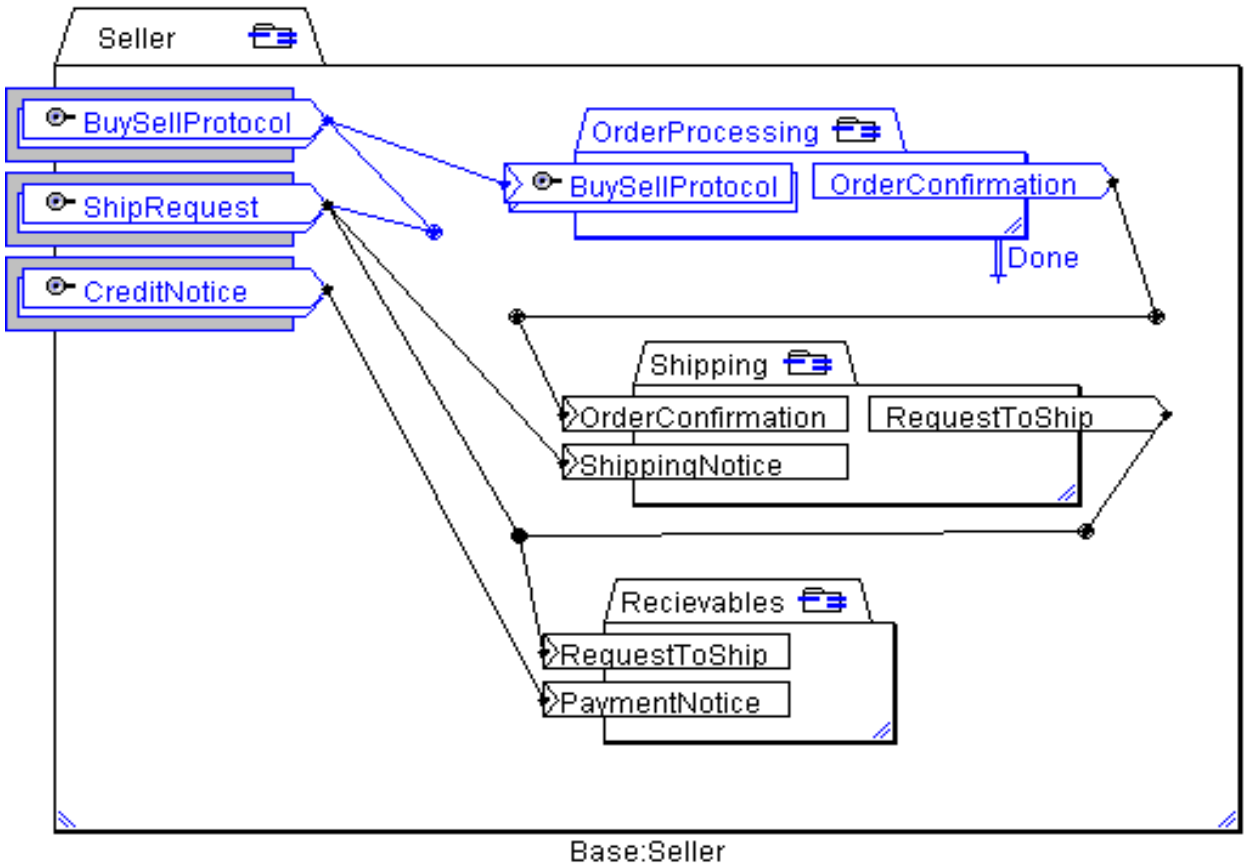




Add ports to complete community process



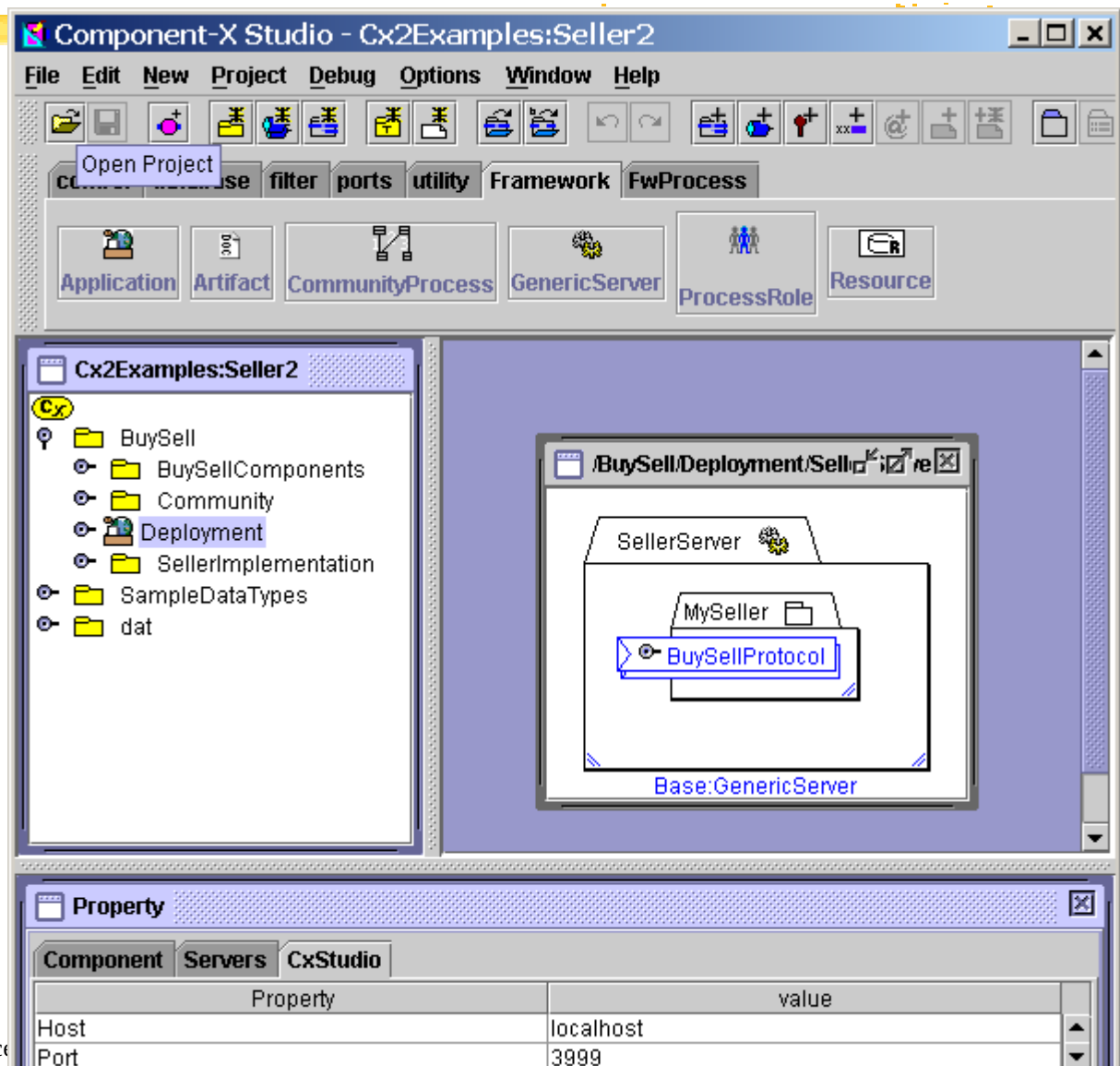
Drill-down



Add implementation

- As component compositions
- In a programming language
- By using an external service

Add technology specifics for deployment



WSEC

Web Services for Enterprise Collaboration

Initial Proposal

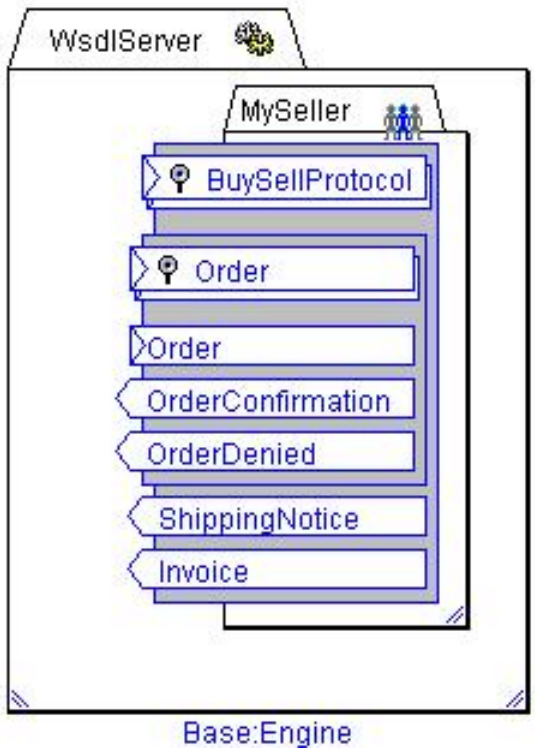
Main Parts

- Distributed Component Profile
  - How to define & structure distributed components
- Aspects
  - How to augment a model for a technology
- Aspects defined for WSDL/Schema
  - The augmentations required for Web Services
- Mapping
  - Transformations between ECA and WSDL/Schema

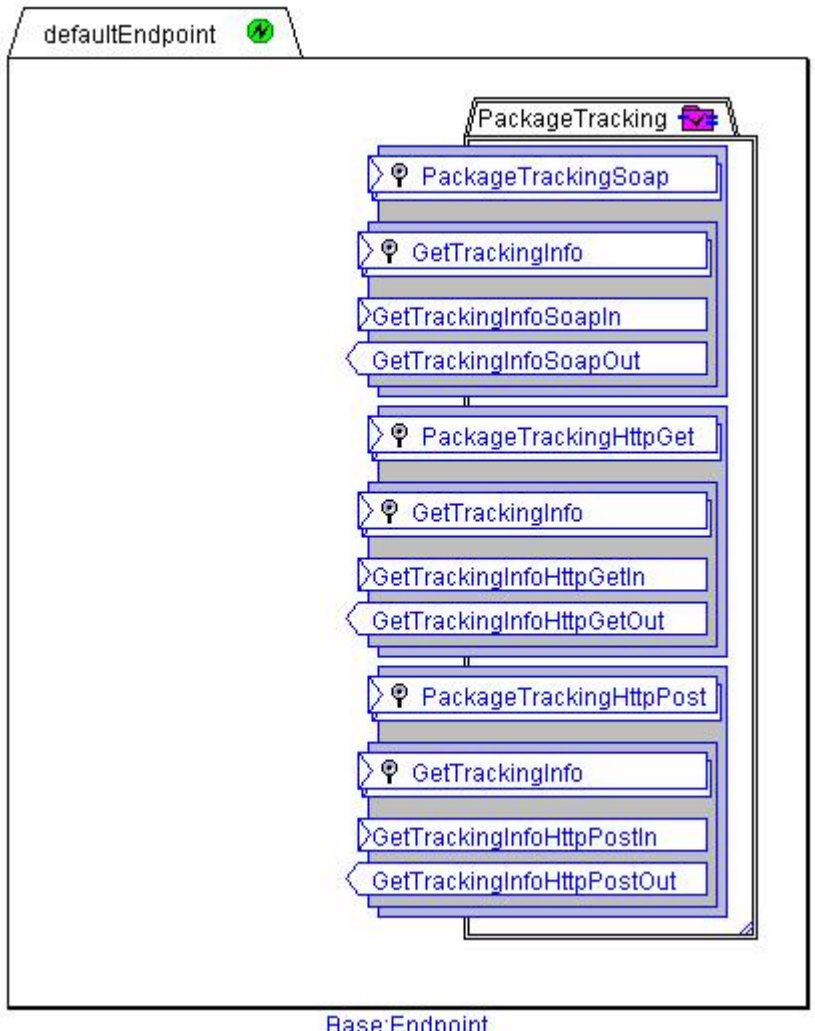
Distributed Components

- Define “role” as the abstract contract
- Define “Engine” as exposing a set of DCs
- Define “Endpoint” as consuming a set of DCs
- Define “Proxy” as the use of an external role

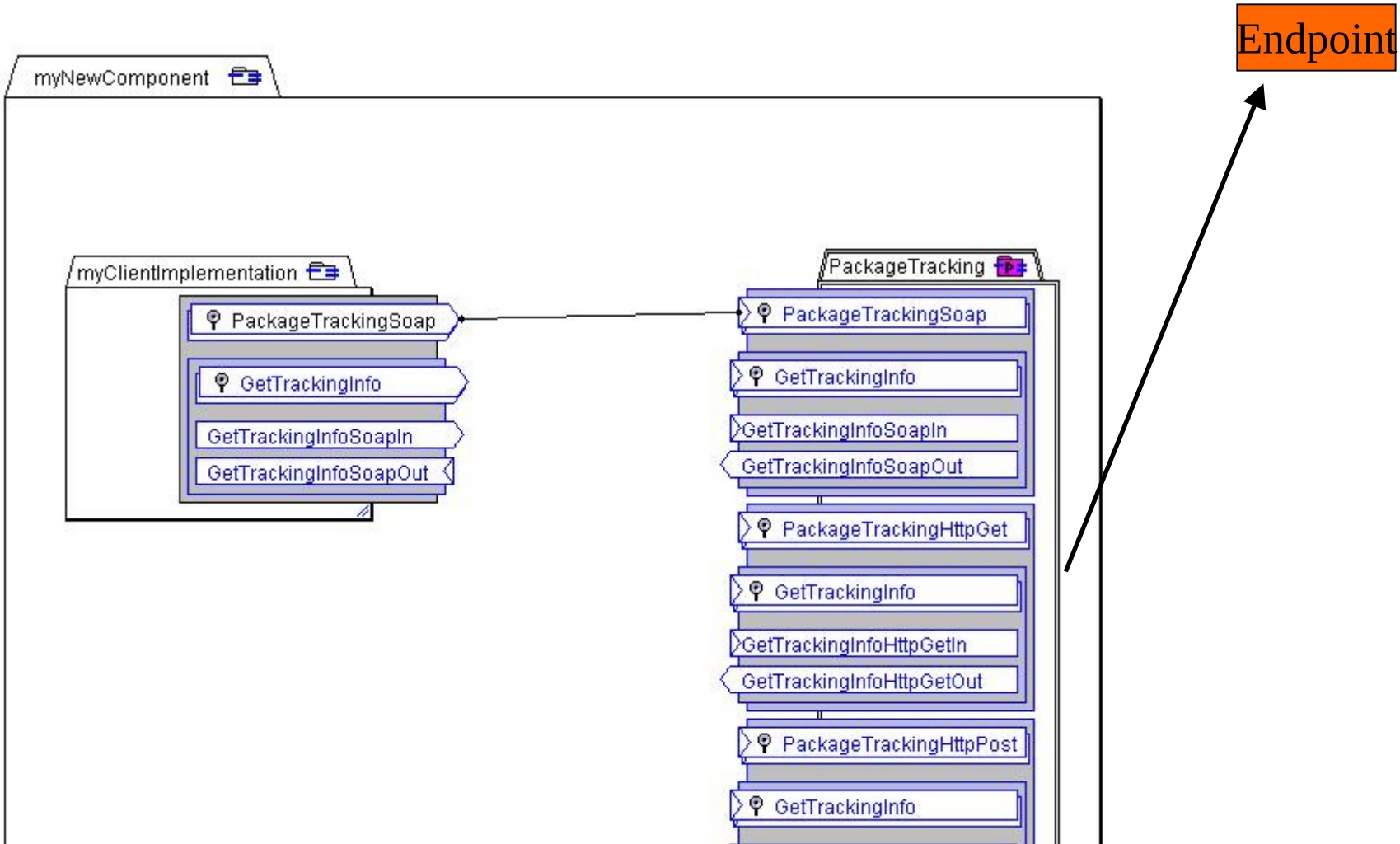
Engine exposing a DC



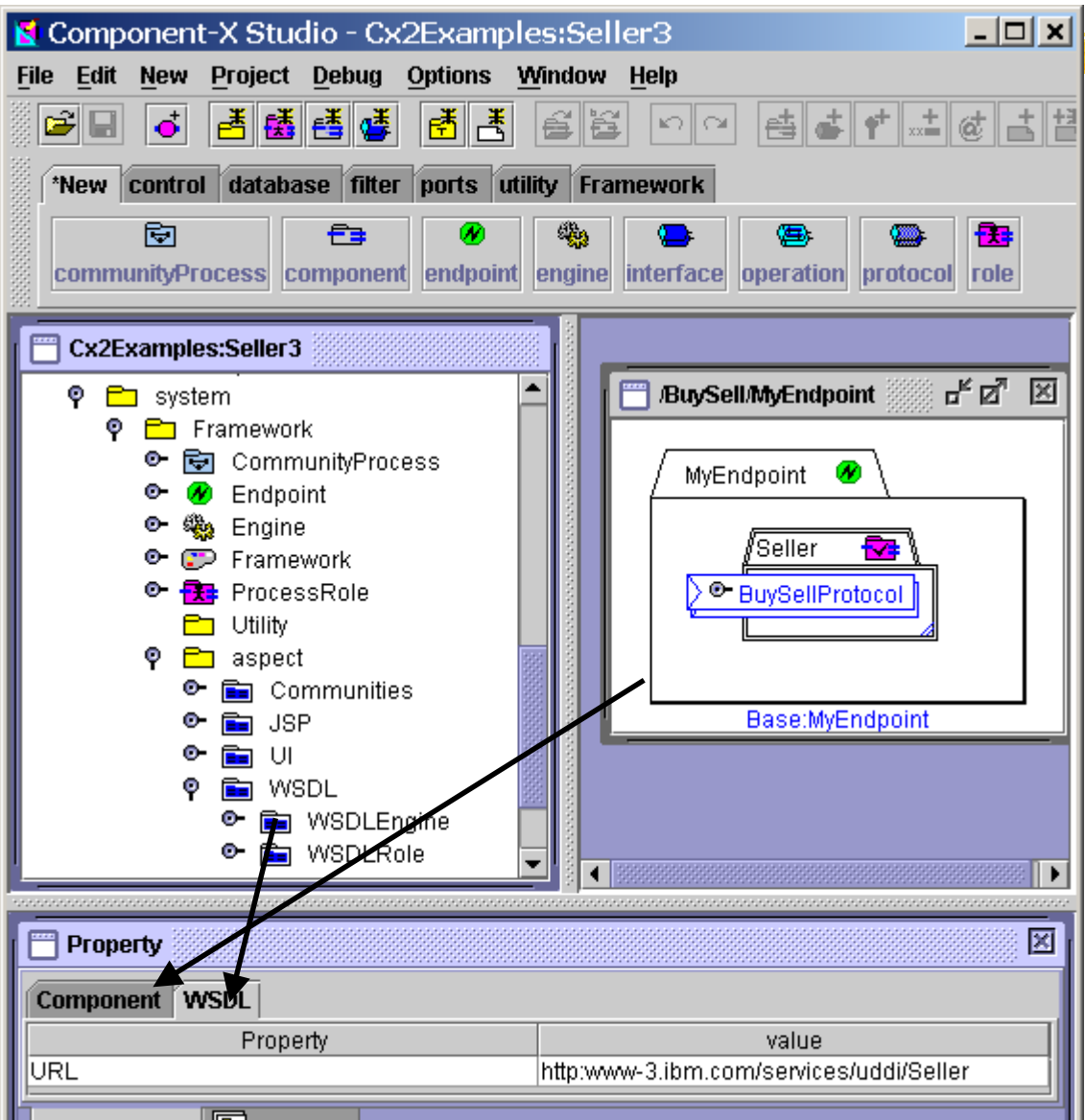
Defining an external component resource



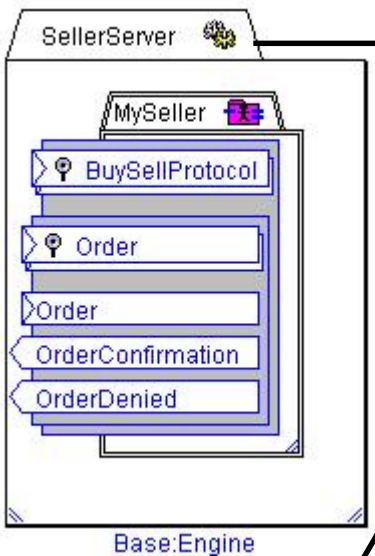
Using a proxy



Aspects



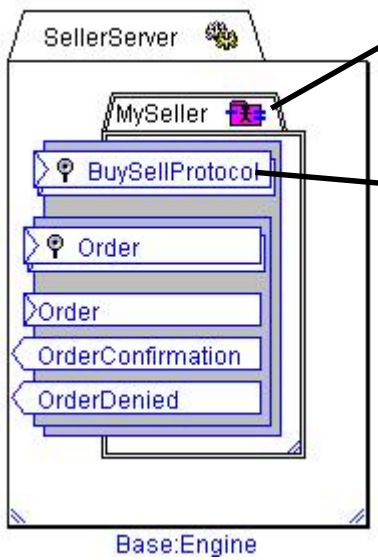
Mapping of a WSDL Engine



- <definitions xmlns="http://schemas.xmlsoap.org/ws
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soa
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mi
xmlns:http="http://schemas.xmlsoap.org/wsdl/http
ENC="http://schemas.xmlsoap.org/soap/encoding/
xmlns:xs2000="http://www.w3.org/1999/XMLSchem
xmlns:xs2001="http://www.w3.org/2001/XMLSchem
targetNamespace="urn:SellerServer" xmlns:tns="urn:S
xmlns:CoreTypes="urn:CoreTypes" xmlns:Ordering="ur
- <!--
definitions obtained from component /BuySell/Deployment/Selle

Aspects
WSDL
WSDL-SOAP

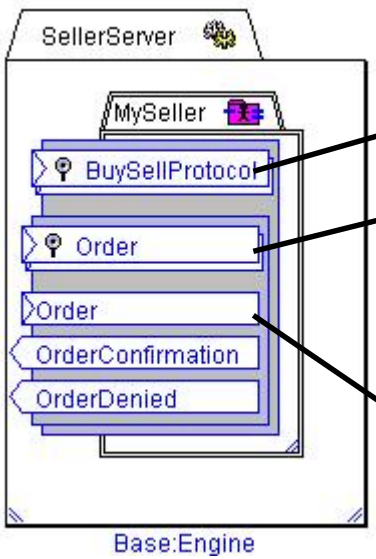
Mapping of a DC



Aspects
WSDL
WSDL-SOAP

```
- <service name="MySeller">
- <!--
    implemented service role
/BuySell/Deployment/SellerServer/MySeller    -->
    <documentation><p> </p></documentation>
  - <port name="BuySellProtocol"
binding="tns:BuySellProtocol">
  - <!--
    original service port was
/BuySell/Deployment/SellerServer/MySeller/BuySellProtoc
ol (extending Component
&lt;/BuySell/SellerImplementation/MySeller/BuySellProtoc
ol&gt; )    -->
    <soap:address
location="http://localhost:8080/cx/app/BuyS
ell/Deployment/SellerServer/MySeller/BuyS
ellProtocol" />
    </port>
```

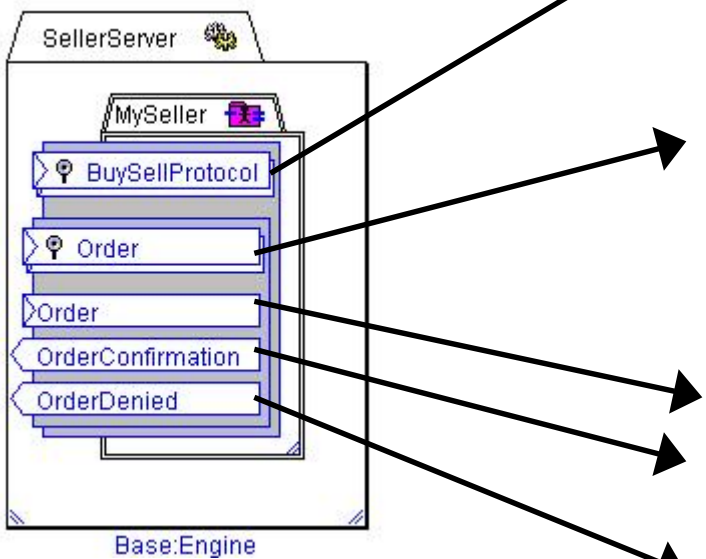
Mapping of a protocol binding



Aspects
WSDL
WSDL-SOAP

```
<binding name="BuySellProtocol"
 type="tns:BuySellProtocol">
<soap:binding
      transport="http://schemas.xmlsoap.org/soap/h
      style="rpc" />
<operation name="Order">
<soap:operation
      soapAction="urn:/BuySell/Community/BuySellP
      col/Order" style="rpc" />
<input name="Order">
<soap:body use="encoded" namespace="urn:SellerServ
      encodingStyle="http://schemas.xmlsoap.org/so
      encoding/" />
```

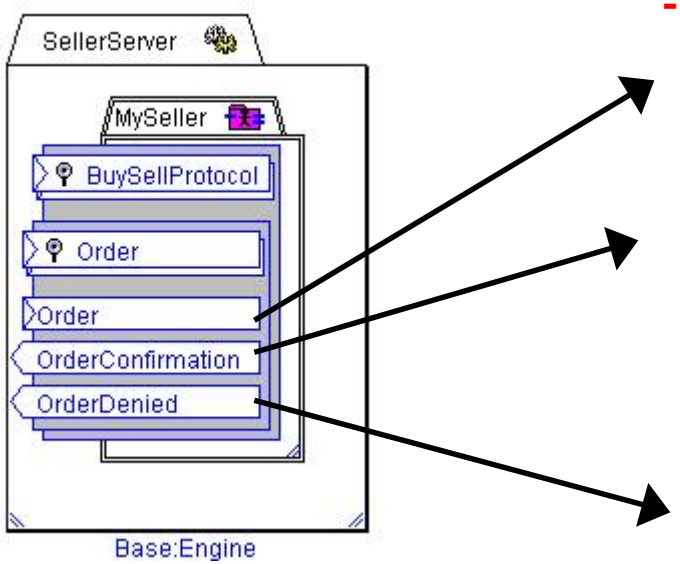
Mapping of a protocol




```
- <portType name="BuySellProtocol">
- <!--
  original cx operation =
/BuySell/Community/BuySellProtocol/Order    -->
- <operation name="Order">
- <!--
  original cx flow port =
/BuySell/Community/BuySellProtocol/Order/Order    -->
  <input name="Order" message="tns:Order" />
  <output name="OrderConfirmation"
message="tns:OrderConfirmation" />
  <fault name="OrderDenied"
message="tns:OrderDenied" />
  </operation>
  </portType>
```

Aspects
WSDL
WSDL-SOAP

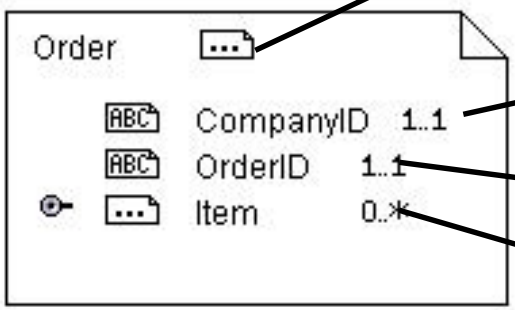
Mapping of message types



Aspects
WSDL
WSDL-SOAP

```
- <message name="Order">
    <part name="Order" type="Ordering:Order"
<message name="OrderConfirmation">
    <part name="OrderConfirmation"
  type="Ordering:OrderConfirmation" />
    </message>    </message>
- <message name="OrderDenied">
    <part name="OrderDenied"
  type="Ordering:OrderDenied" />
    </message>
```

Mapping of data types

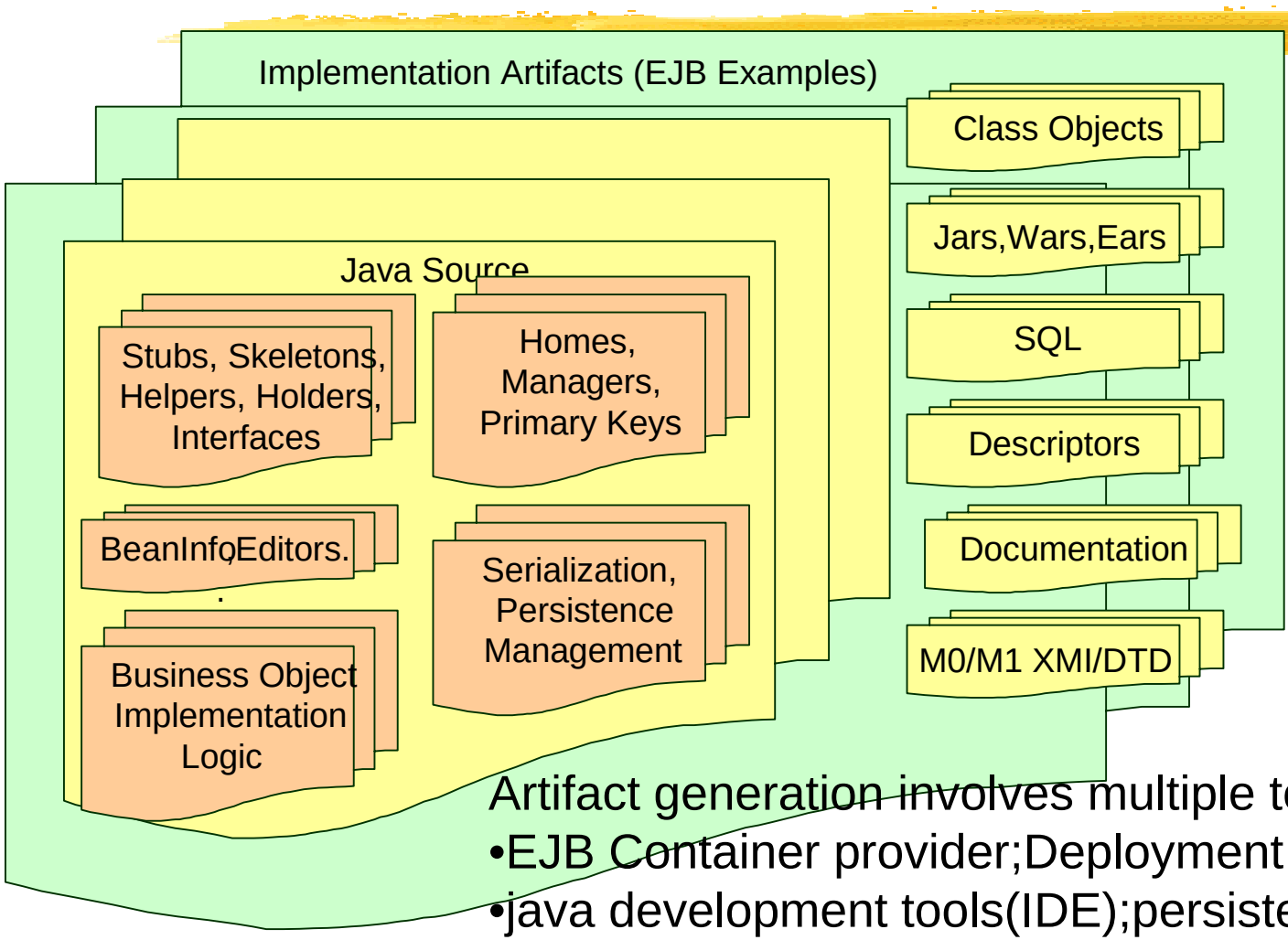



```
- <xs2001:complexType name="Order">
-
- <xs2001:sequence>
    <xs2001:element minOccurs="1"
  maxOccurs="1" name="CompanyID"
  type="CoreTypes:CompanyID" />
    <xs2001:element minOccurs="1"
  maxOccurs="1" name="OrderID"
  type="Ordering:OrderID" />
    <xs2001:element minOccurs="0"
  maxOccurs="unbounded" name="Item"
  type="Ordering:Item" />
    </xs2001:sequence>
    </xs2001:complexType>
```

Ways to map

- Generate code and other artifacts
  - Usually required to adapt to legacy
  - Emulates the manual process
- Assemble and configure existing generic and specific components
  - Both efficient and dynamic
- Interpret or otherwise “animate” the high-level model
  - High level & dynamic but may have performance issues
- Simulate
  - Evaluate technical and business impact through simulation

Example Generated Artifacts



Artifact generation involves multiple tools

- EJB Container provider;Deployment tools;Packagers;
- java development tools(IDE);persistence provider;…

Typical 10-20 per PIM Classifier

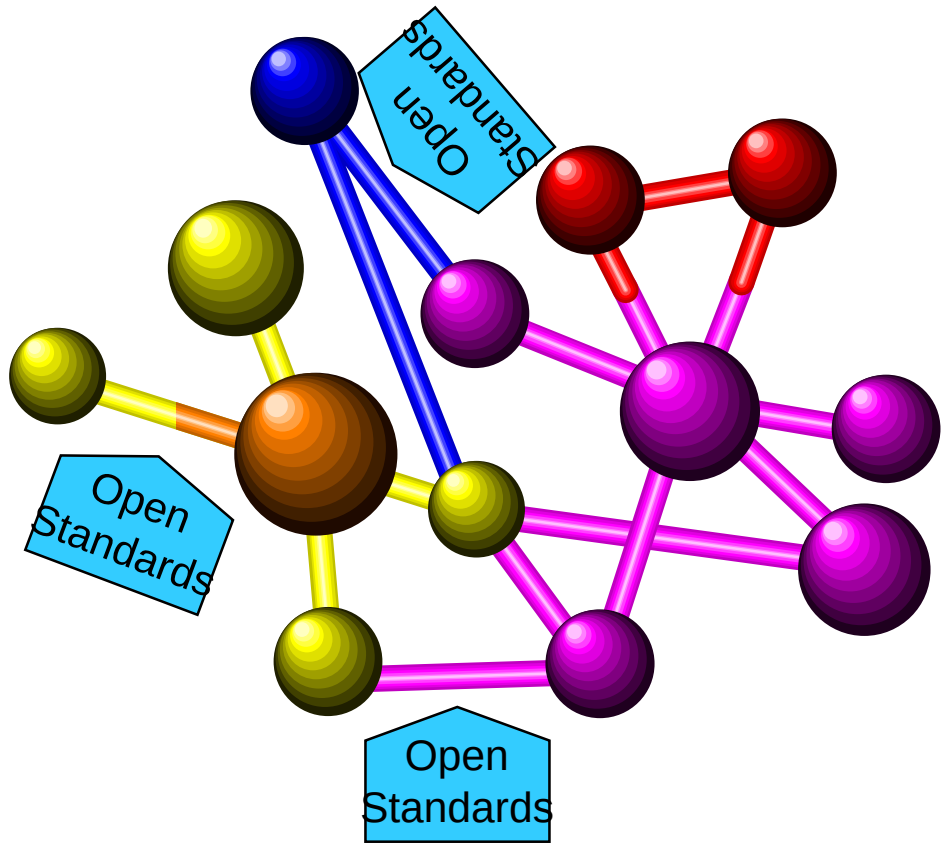
0-20% manual override

Vision

Building and adapting systems for collaboration, reuse and change

Loosely coupled enterprise architecture

- Independent enterprise components
- Representing business concepts
- Link via open standards

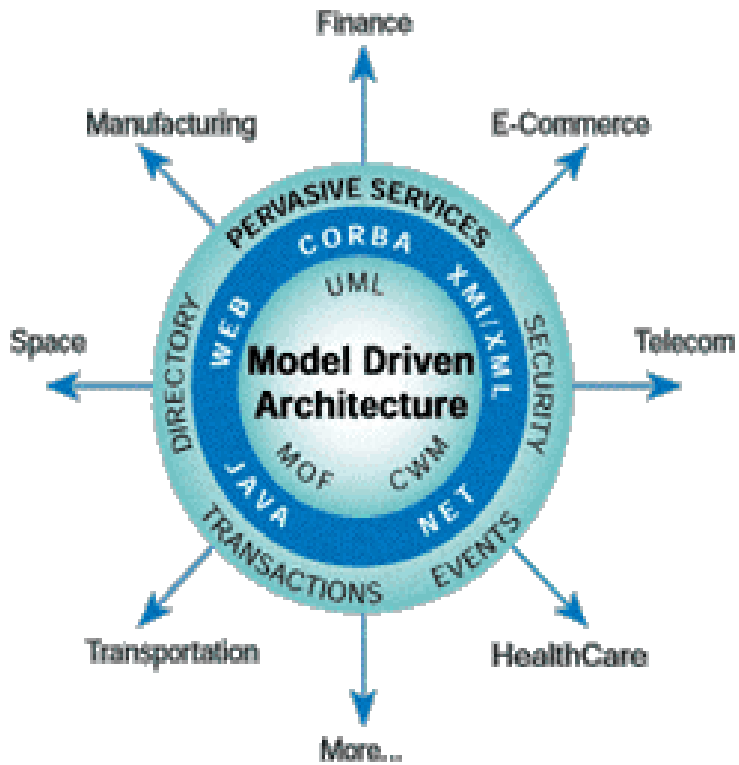


Business Component Marketplace

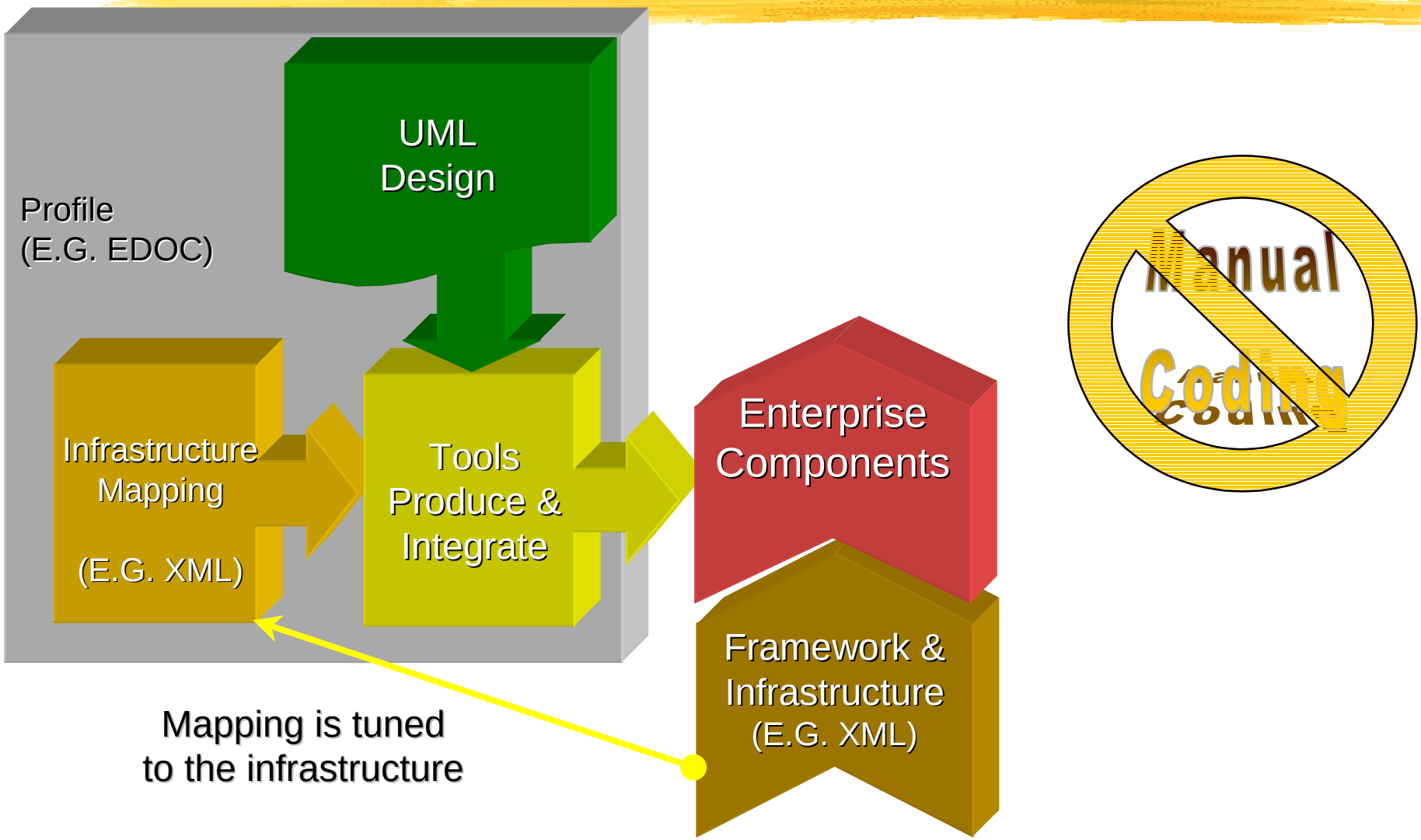
- The business component marketplace is projected to be a 10b market in 5 years
- Consider the value of XML components that wrap popular legacy
- New application functionality built from components
- Components for integration and transformation
- XML and web services makes an excellent basis for such components
- Technology components, such as for repositories and DBMS
- Marketplace my be inside the enterprise or commercial

OMG Model Driven Architecture (MDA)

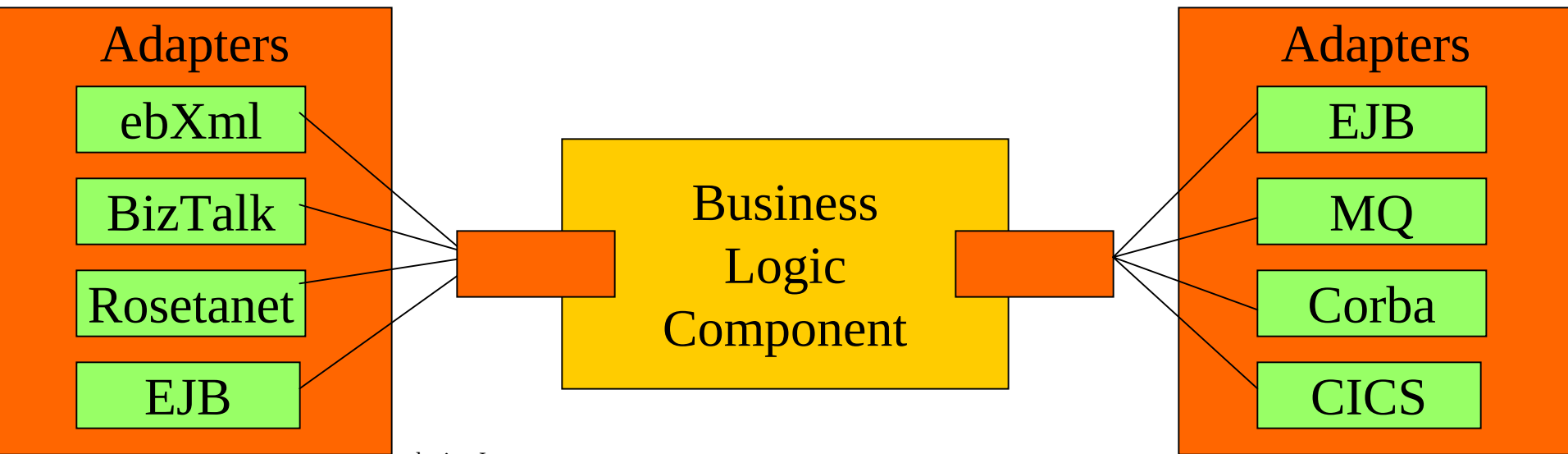
- High level – platform independent models
- Technology Models
- Mapping
  - Custom
  - Standard
- Standard Models produce technology specific standards artifacts



Automated MDA



Technology Independence

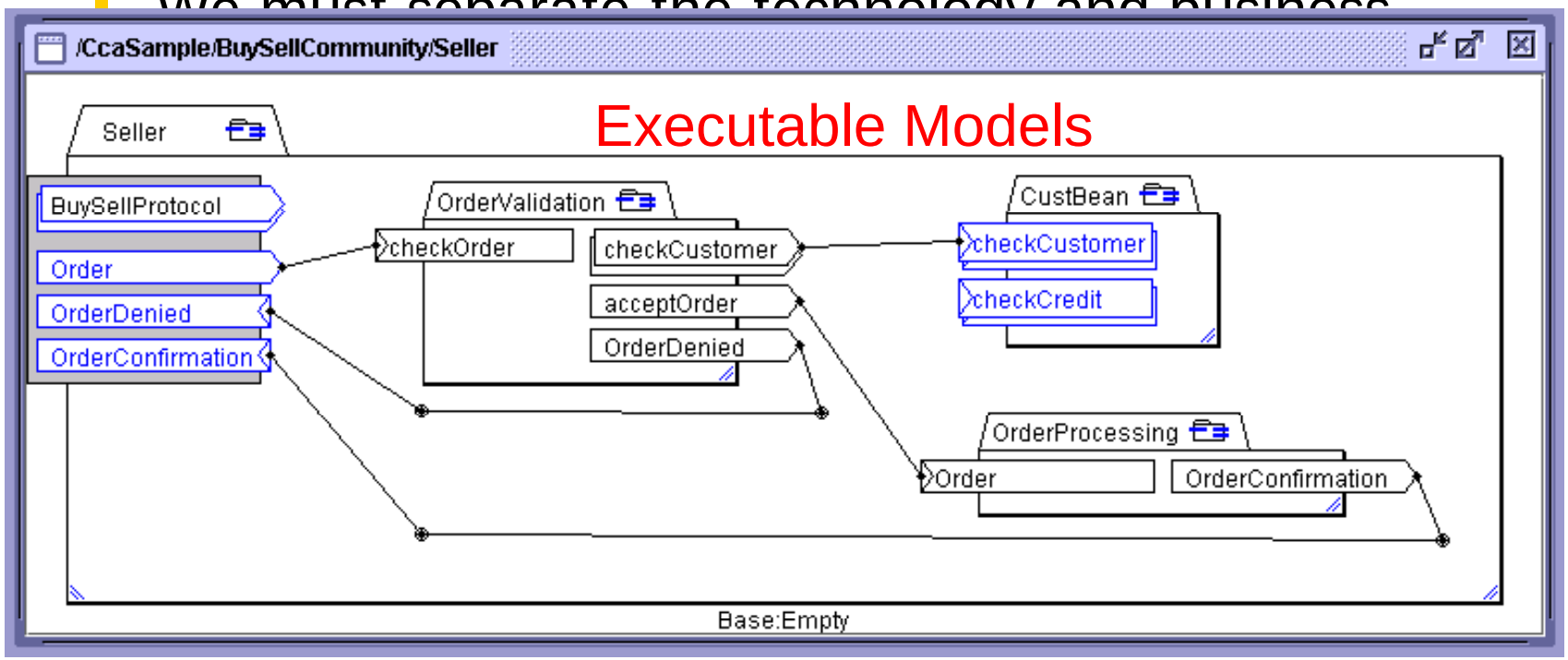


High level tooling & infrastructure

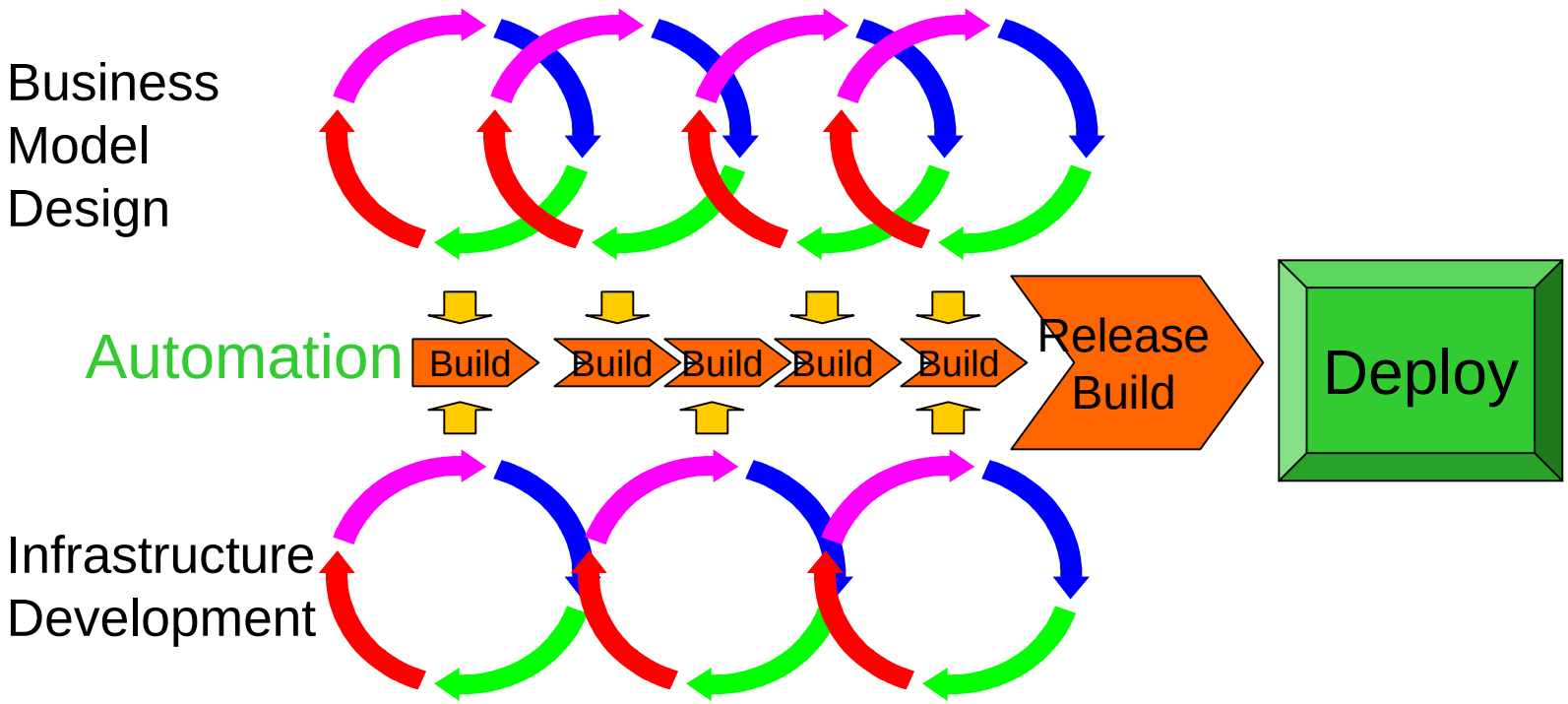
- MUST BE SIMPLE!
  - We must be able to create better applications faster
  - We must separate the technology and business concerns, enable the user
- Tooling + Infrastructure
  - Executable models are source code
  - Tooling must be technology aware
  - Infrastructure must support tooling, not manual techniques
- Model based component architectures

High level tooling & infrastructure

- MUST BE SIMPLE!
  - We must be able to create better applications faster
  - We must separate the technology and business



Iterative Development



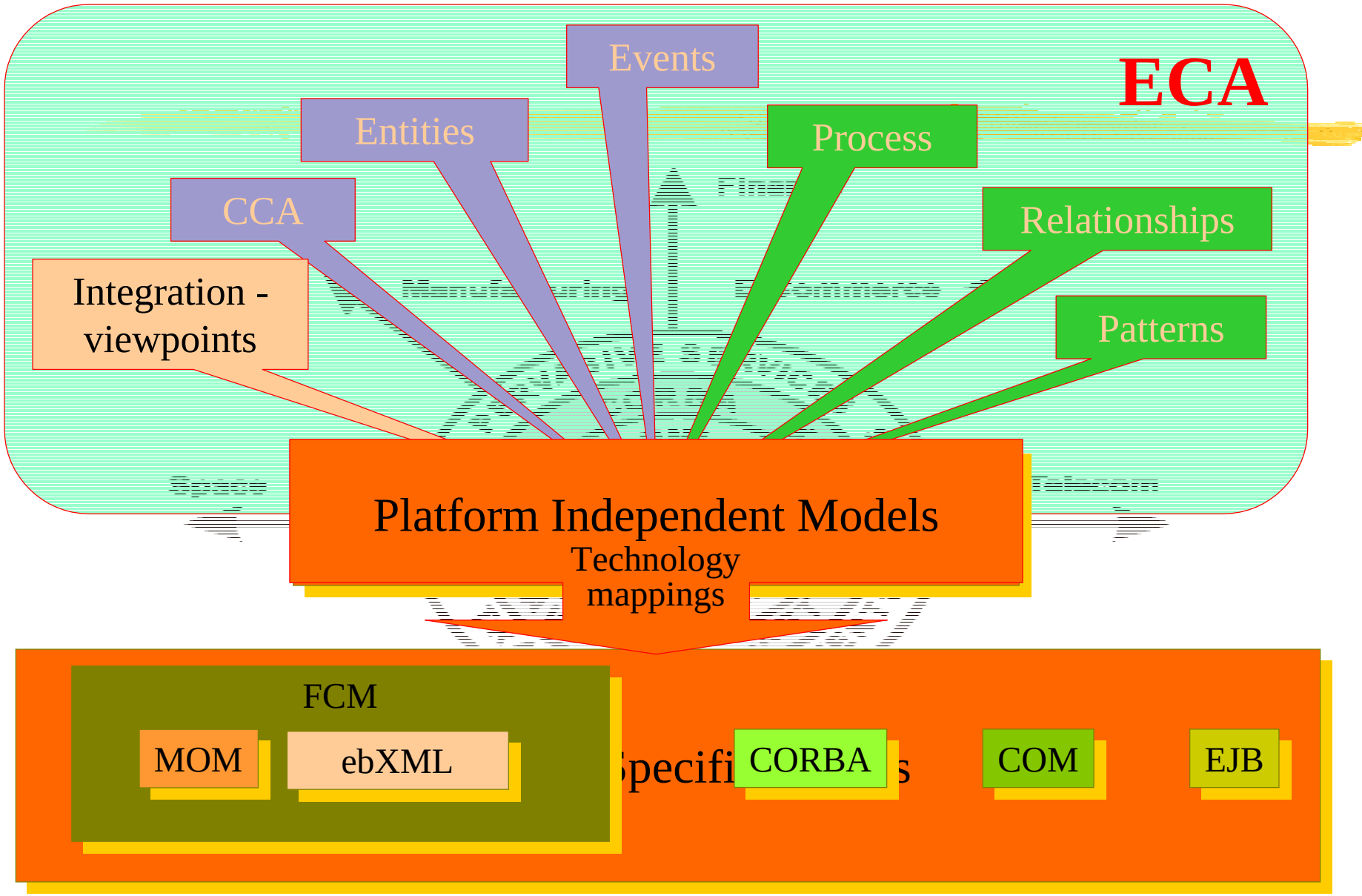
MDA Solution Factory

- Put together the
  - Best practices
  - Expertise
  - Enterprise Architecture
  - Infrastructure
  - Automated tooling
- To produce and integrate robust business collaborations quickly & reliably

Net effect

- Using these open standards and automated techniques we can;
  - Achieve the strategic advantage of an open and flexible enterprise
  - Produce and/or integrate these systems FASTER and CHEAPER than could be done with legacy techniques
  - Provide a lasting asset that will outlive the technology of the day

Advanced EDOC



Business Process Profile

how things are coordinated

Business Processes

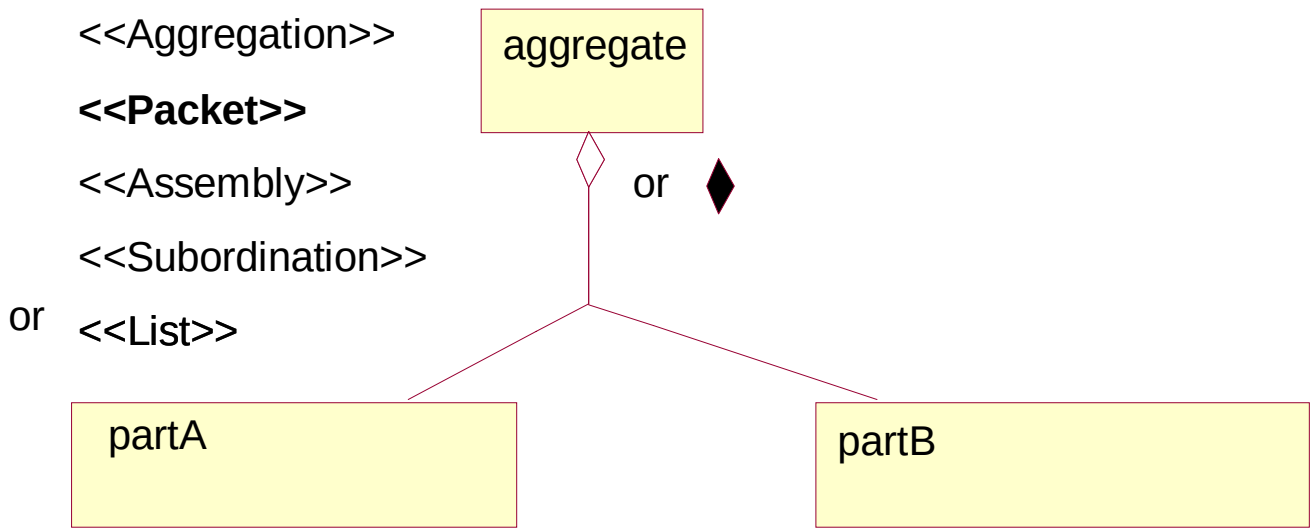
- Specialize CCA
- Activity-centric view of a Process
- Express
  - Complex temporal and data dependencies between business activities
  - Iteration of activities
  - Alternative required Inputs and Outputs of activities
  - Roles related to performers, artifacts and responsible parties for activities

Relationships Profile

Useful associations and dependencies

Relationships Profile

- Enables non-binary aggregations
- Defines useful association and dependency stereotypes



Patterns Profile

reusing
parameterised
designs

Patterns Profile

- Profiles UML Parameterized Collaborations
- Based on Business Function Object Patterns (BFOP)
  - Multi-Layer
  - Based on Catalysis Approach
- Adds stereotypes for
  - Named Patterns
  - Inheritance
  - Composition
  - Pattern Binding with renaming

Platform Specific Modelling

- EJB, FCM, MOF
- Technology mappings from EDOC to Distributed Component and Message Flow Platform Specific Models
  - EDOC to J2EE/EJB mapping
  - EDOC to CORBA/CCM mapping
  - EDOC Business Process to FCM mapping
  - EDOC Business Process to CORBA mapping

Discussion

- How much are we using these kinds of concepts today?
- What would be the advantages to going in this direction?
- What are the barriers;
  - Technical?
  - Political?
- What would be a good first step?

Contact

Cory Casanave

Data Access Technologies

www.enterprise-component.com

cory-c@enterprise-component.com

(305) 234-7077