

Web Services:

Distributed Applications without Limits

- An Outline -

Keynote at BTW2003 & KiVS2003
(Leipzig, Germany, 26. February 2003)

Prof. Dr. Frank Leymann
IBM Distinguished Engineer
Member, IBM Academy Of Technology

IBM Software Group
Schoenaicherstr. 220
71032 Boeblingen
Germany

Phone	+49-7031-16 3998
Fax	+49-7031-16 4890
Cellular	+49-172-731 5858
e-mail	Leyl@de.ibm.com

Copyright Notice:

Most of the material of this presentation is copyrighted by IBM or by others. Please don't use any of the material of this presentation without checking with the author mentioned on the front page. Thanks!

Agenda

Introduction
Virtual Components
Virtual Environments
Application Structure
Aggregations
Summary & Conclusion

Agenda

Introduction

Virtual Components

Virtual Environments

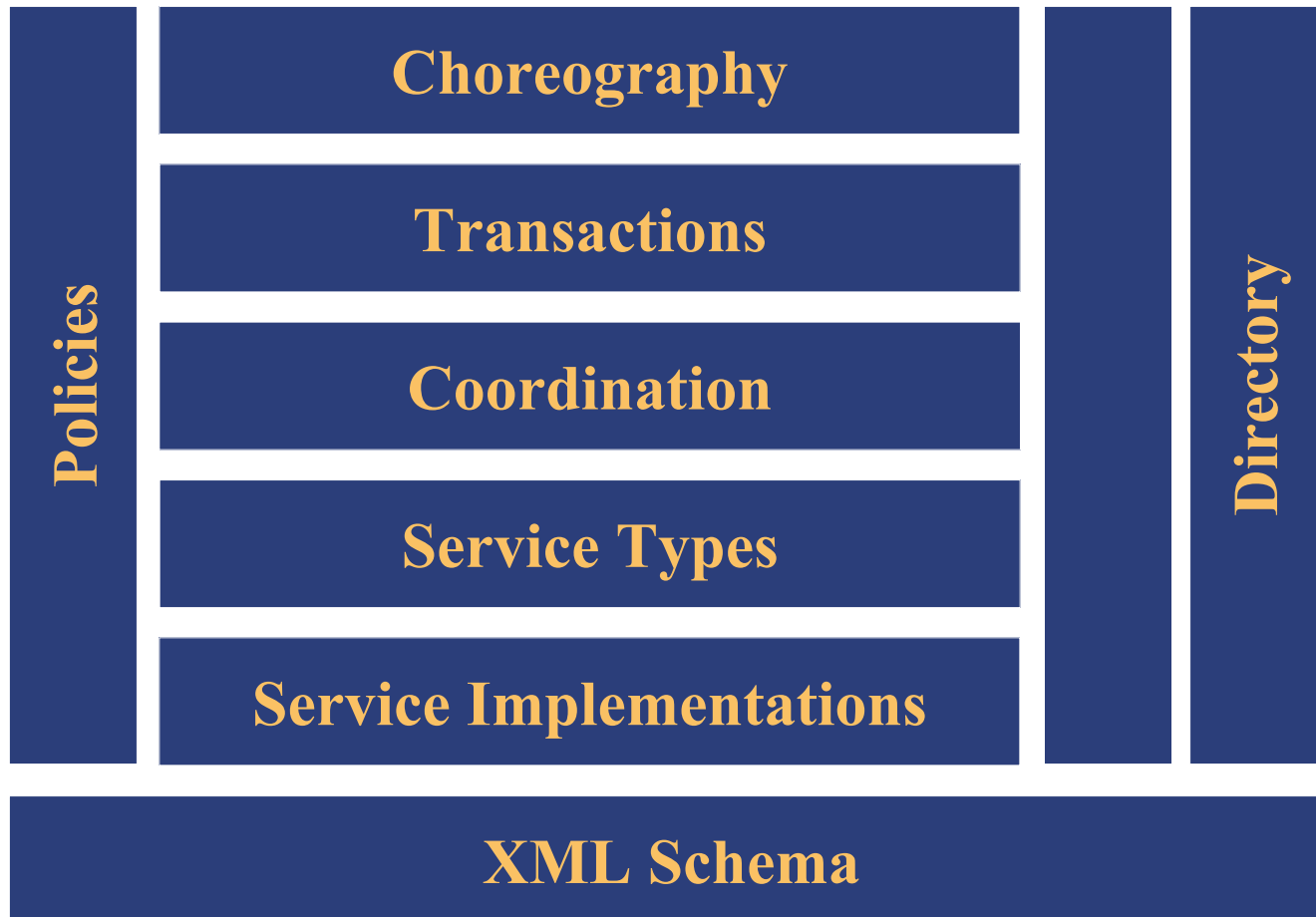
Application Structure

Aggregations

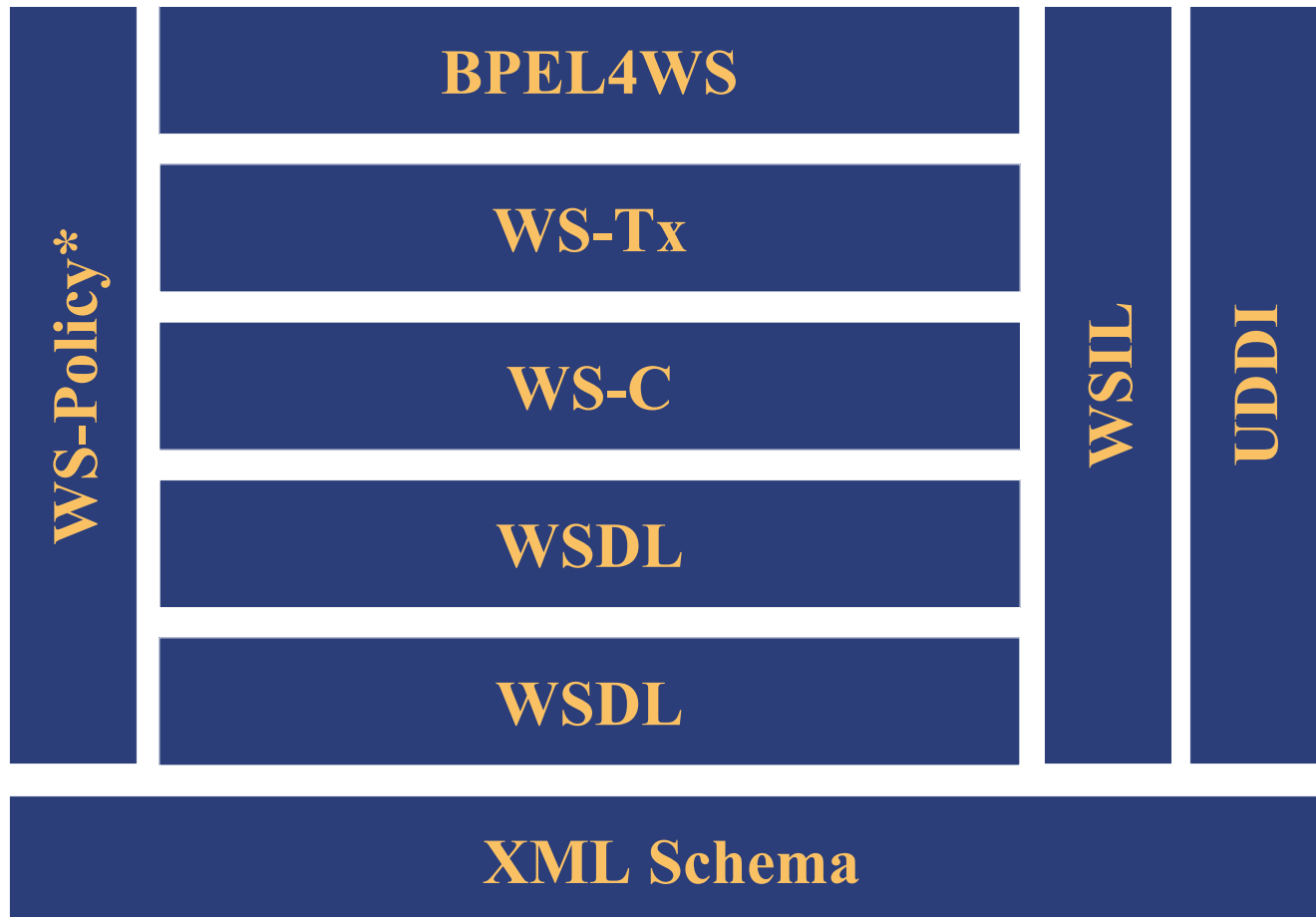
Summary & Conclusion

What's going on
in terms of standards?

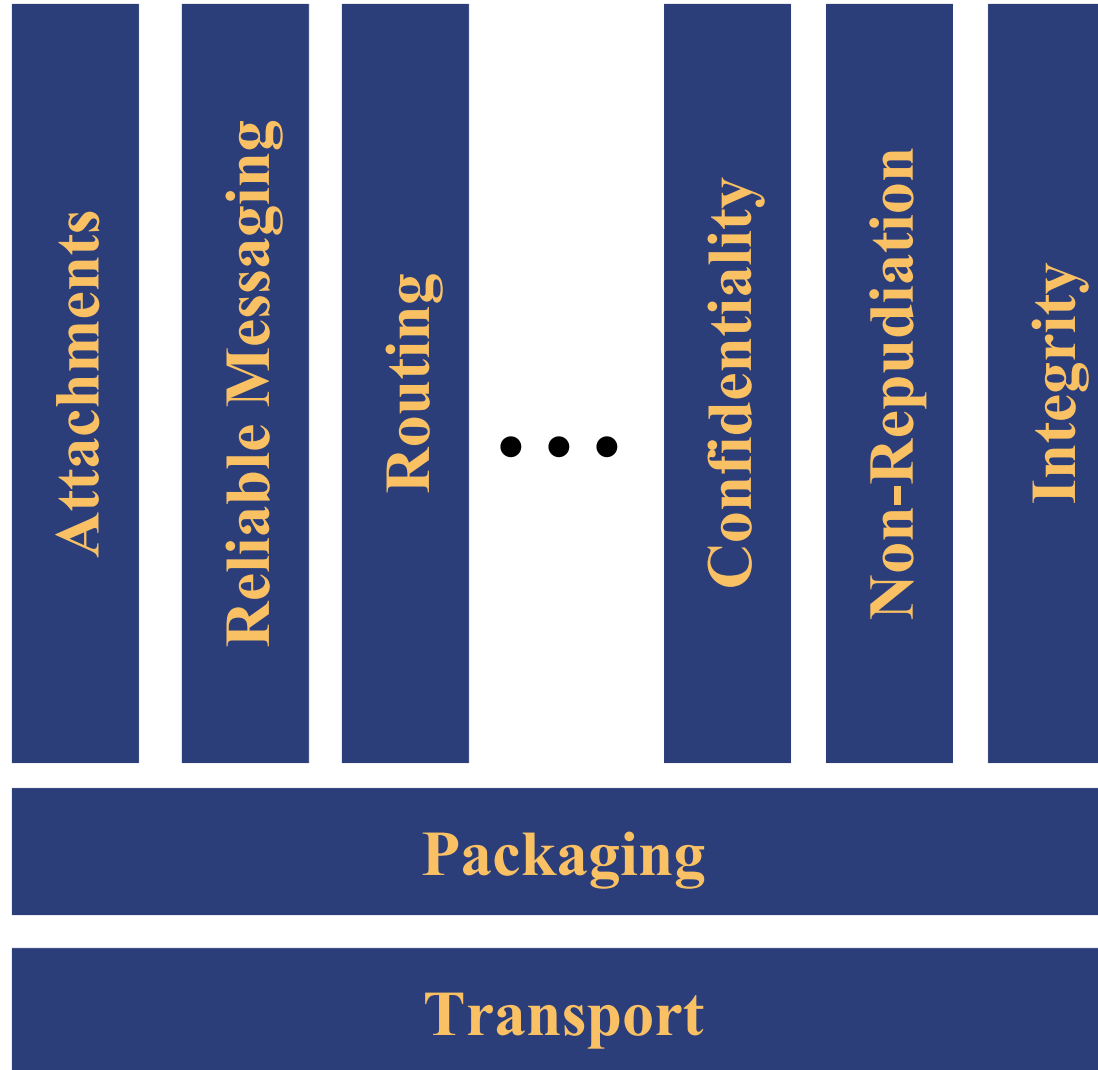
The Description Stack



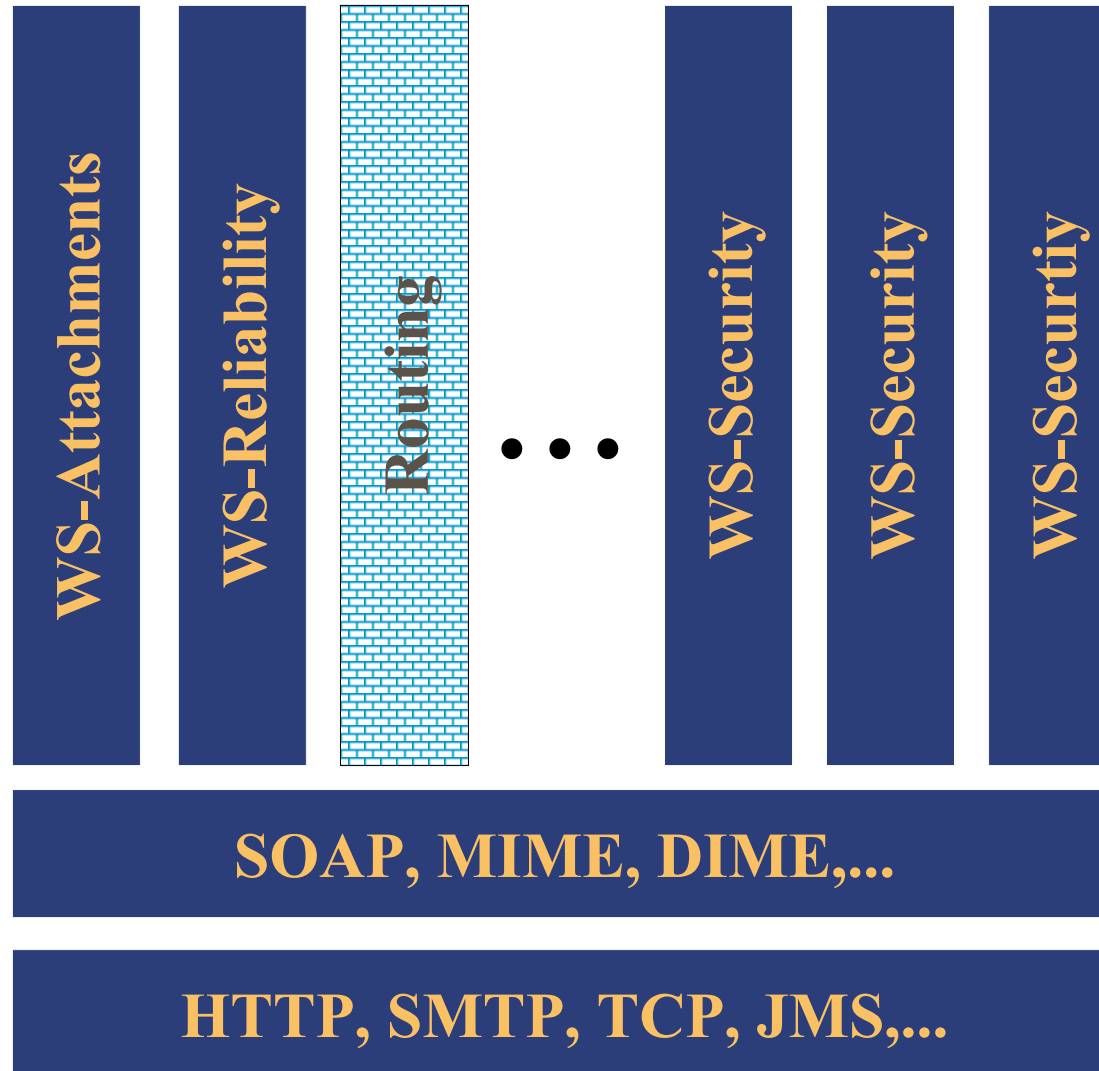
The Description Stack: Specifications



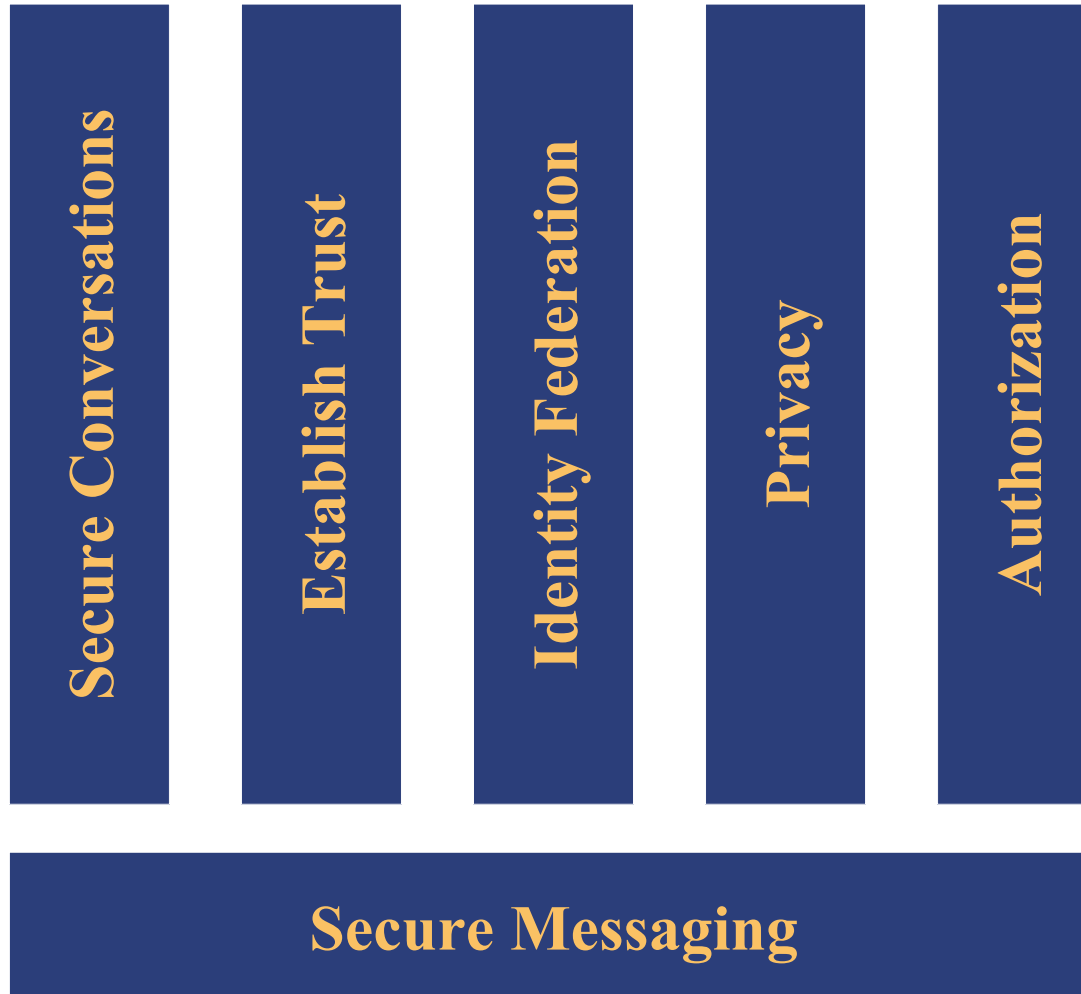
The Wire Stack



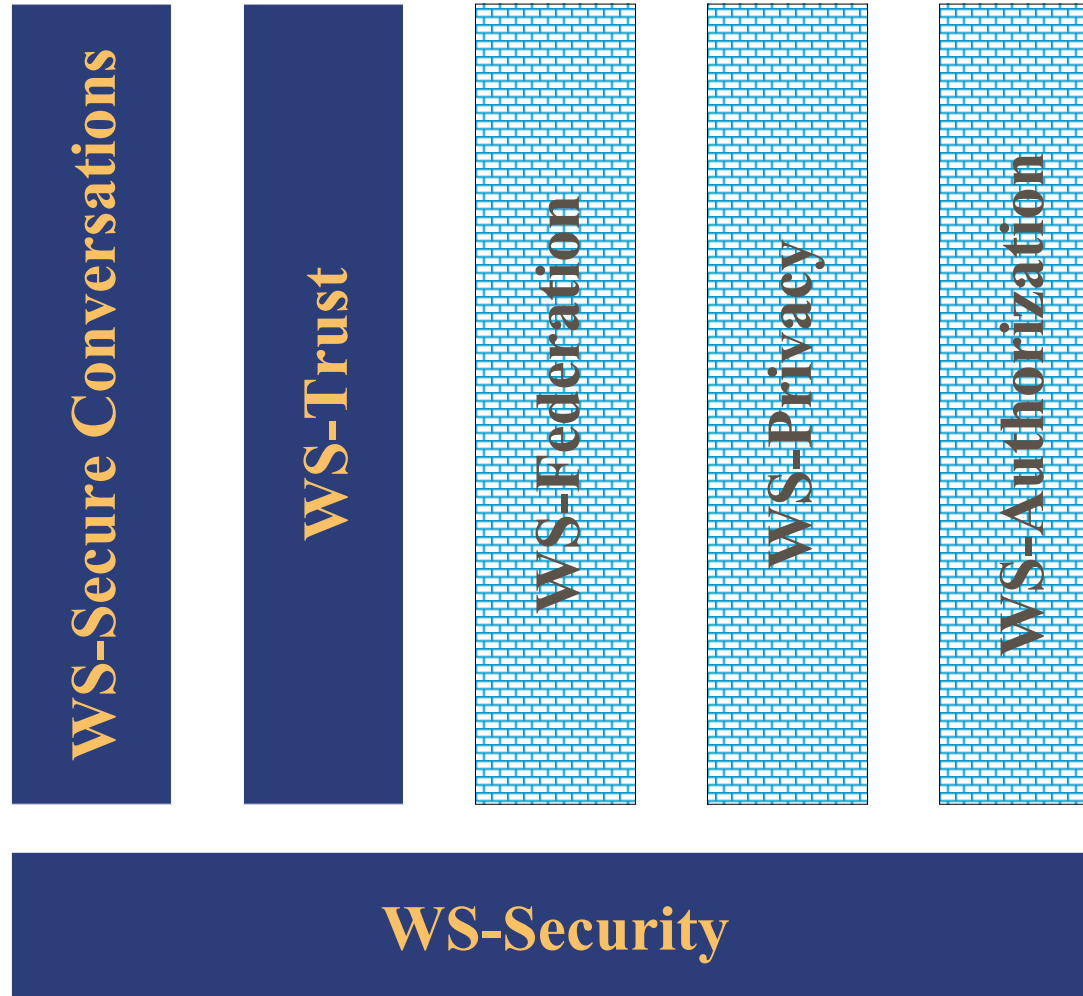
The Wire Stack: Specifications



The Security Stack



The Security Stack: Available Specs



Additional Specifications

- Presentation
 - WSRP
- Service Level Agreements
 - WSLA
- Quality of Services
 - Application domain specific policies
- Management
 - Versioning, compatibility, change management,...
- ...

Interoperability

- WS-I.org (Web Services Interoperability)
 - Consortium to promote interoperability based on “profiles”
- Profile
 - Set of Web services standard specifications plus clarifications to those specification and their usage
- Self-certification of profile interop compliance
- Sample implementations and implementation guidelines
- Test tools (sniffers, analyzers,...)

Agenda

Introduction

Virtual Components

Virtual Environments

Application Structure

Aggregations

Summary & Conclusion

Out of the Galaxy: What Are Web Services At All?

Fundamental Importance

**Web Services bring the benefits of
EDI to everybody!**

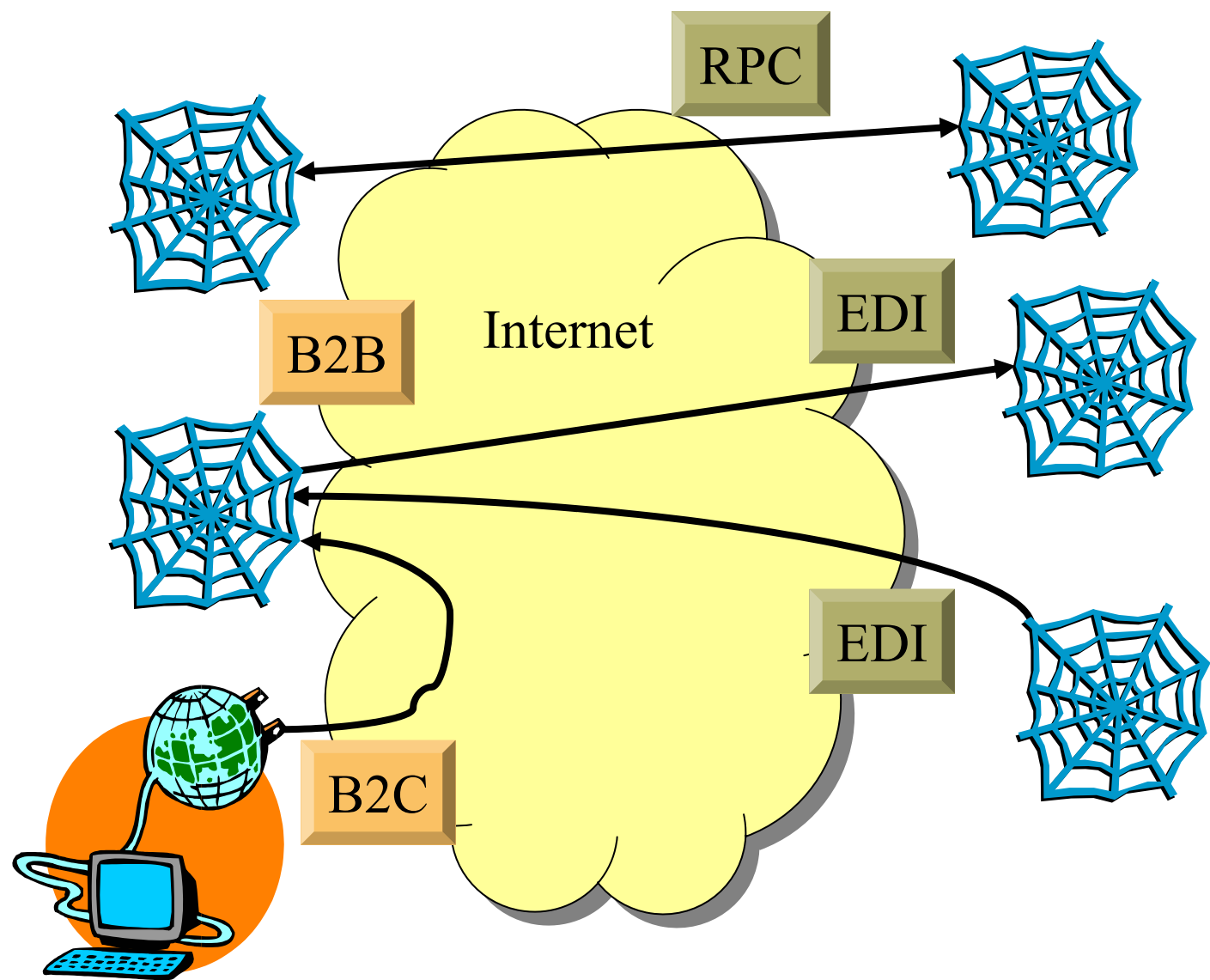
**„Web Services are for B2B
what browsers are for B2C!“**

So, What Is A „Web Service“?

Originally.....

**A Web Service is
any piece of code
that can communicate with
other pieces of code
via common Internet technology**

Web Services Usages

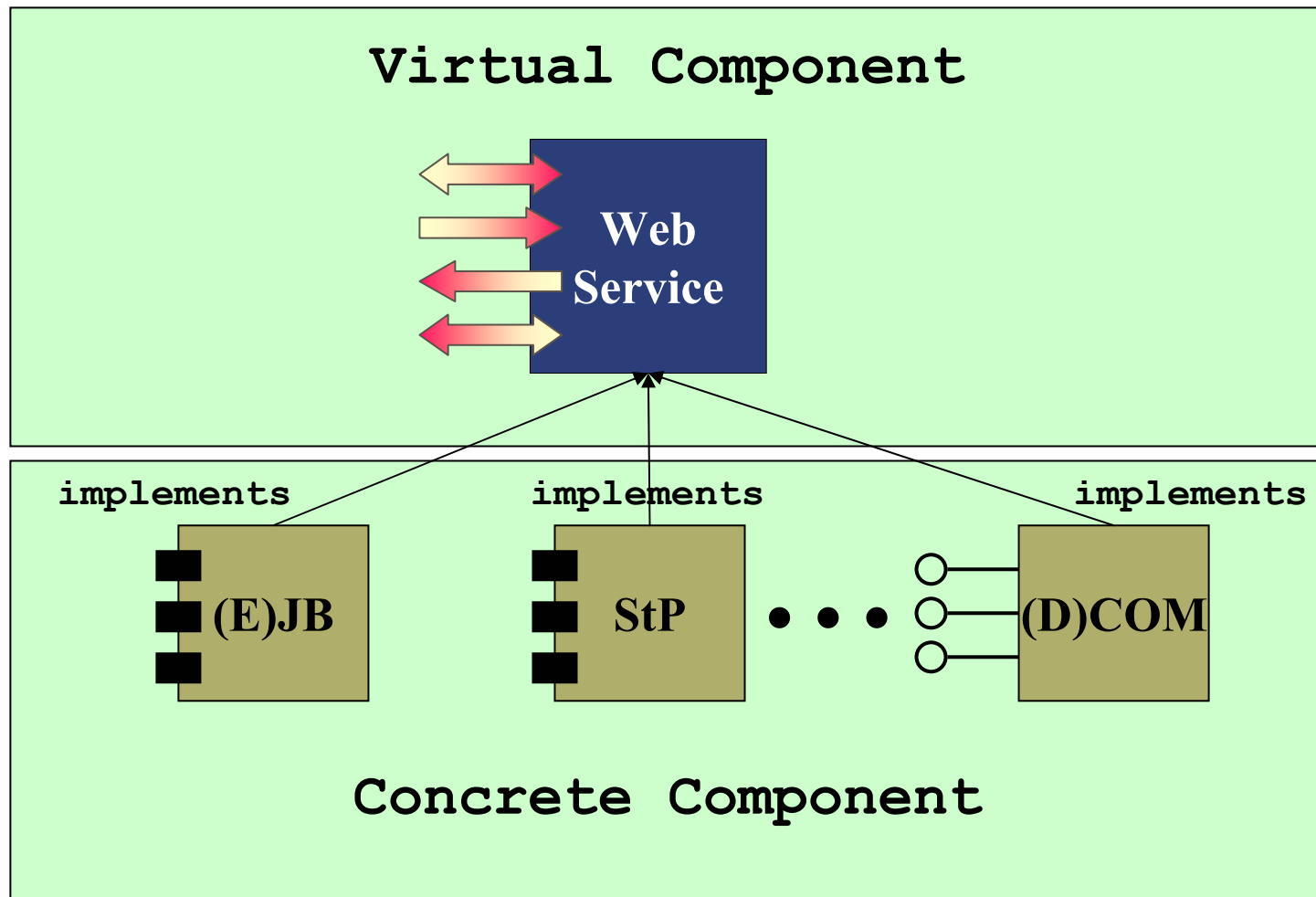


Again: What Is A „Web Service“?

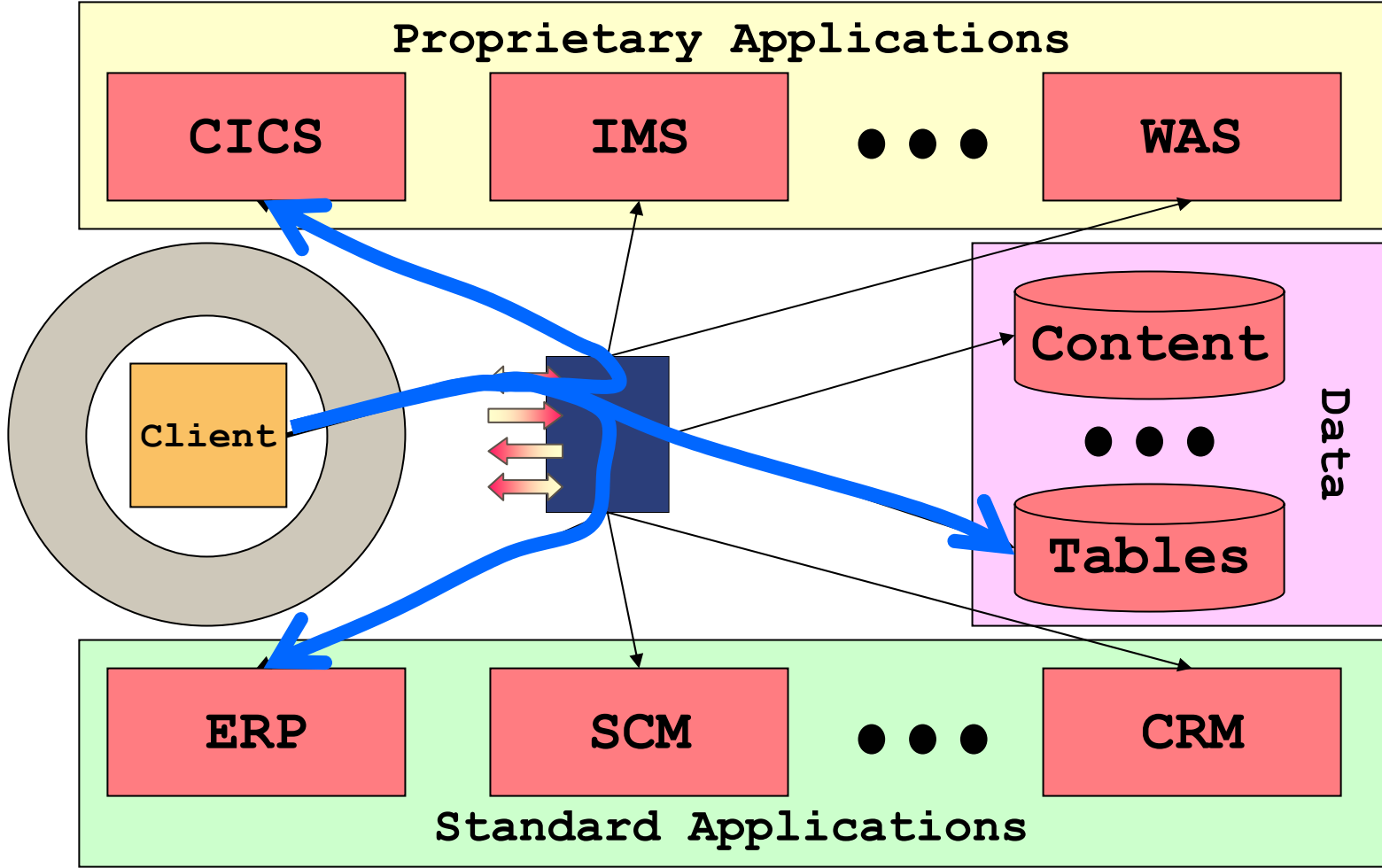
...but in the meantime...

**A Web Service is a
„virtual component“ that hides
„middleware ideosyncracies“
like the underlying component
model, invocation protocol etc.
as far as possible.**

Virtualizing Components

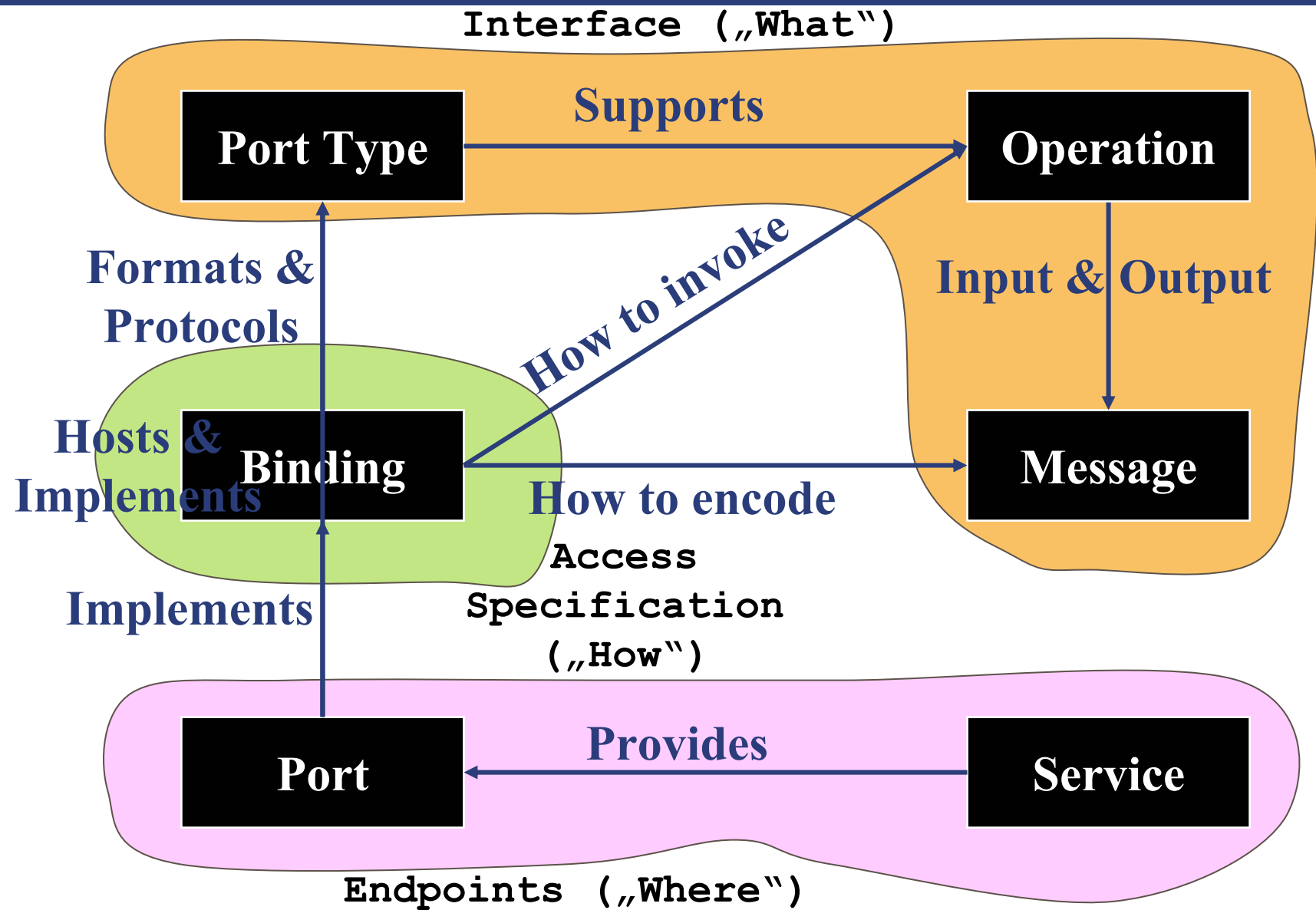


Again: Web Services Usages



WSDL In A Nutshell

Ingredients Of WSDL



Port Type: Example

```
<?xml version="1.0"?>
<definitions name="StockQuote" ... />

  <portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
      <input message="tns:GetLastTradePriceRequest"/>
      <output message="tns:GetLastTradePriceResponse"/>
    </operation>
  </portType>
</definitions>
```


Port & Binding: Example - SOAP/HTTP

```
<binding name="StockQuoteSoapBinding"
  type="tns:StockQuotePortType">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetLastTradePrice">
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
    </input>
    <output>
      ...
    </output>
  </operation>
</binding>

<service name="StockQuoteService">
  <port name="StockQuotePort"
    binding="tns:StockQuoteSoapBinding">
    <soap:address location="http://leymann.com/stockquote"/>
  </port>
</service>
```

Port & Binding: Example - SOAP/JMS

```
<binding name="StockQuoteSoapBinding"
        type="tns:StockQuotePortType">
  <soap:binding style="rpc"
                transport="http://schemas.xmlsoap.org/soap/jms"/>
    (...)
</binding>

<service name="StockQuoteService">
  <port name="StockQuotePort"
        binding="tns:StockQuoteBinding">
    <jms:address destinationType="queue"
                jndiConnectionFactoryName="myQCF"
                jndiDestinationName="myQ"
                initialContextFactory= "com.ibm.NamingFactory"
                jndiProviderURL= "iiop://something:900/" />
  </port>
</service>
```

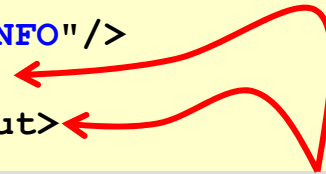
Port & Binding: Example - EJB Invocation

```
<binding name="DefaultPortTypeJavaBinding" type="tns:DefaultPortType">
  <ejb:binding/>
  <operation name="readCustomer">
    <ejb:operationmapping method="getCustomer"
      partOrdering="firstName lastname customerNo"
      interface="remote" />
    <input name="RequestMessage"/>
    <output name="ResponseMessage"/>
    <fault name="FaultMessageCustomerNotFound"/>
    <fault name="FaultMessageInvalidCustomerNumber"/>
  </operation>
</binding>

<service>
  <port name="DefaultPortTypeJavaPort"
    binding="tns:DefaultPortTypeJavaBinding">
    <ejb:address class="com.ibm.example.EjbCustomerManager"
      jndiName="myapp/EjbCustomerManager" />
  </port>
</service>
```

Port & Binding: Example CICS via J2C

```
<binding name="MyConnectorBinding" type="tns:MyConnectorPortType">  
  <cics:binding/>  
  <operation name="getCustomerInfo">  
    <cics:operation functionName="CUSTINFO"/>  
    <input name="Request"> ... </input>  
    <output name="Response"> ... </output>  
  </operation>  
</binding>
```

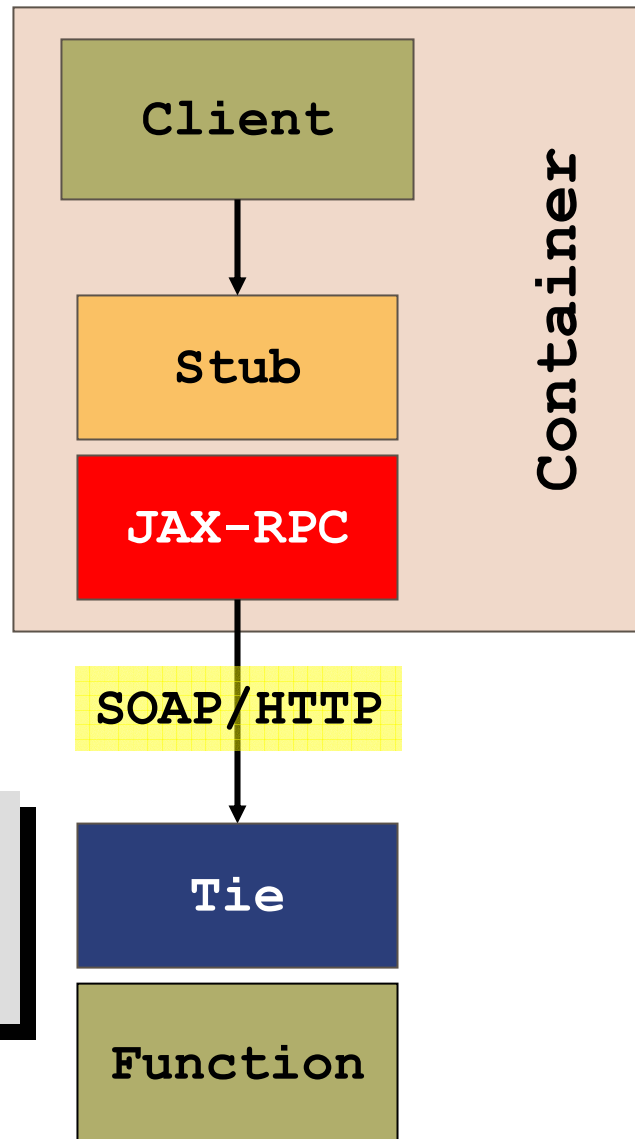


Defining structure of COMMAREA

```
<service name="MyConnectorService">  
  <port name="MyConnectorPort" binding="tns:MyConnectorBinding">  
    <cics:address connectionURL="xyz.ibm.com" serverName="cics21"/>  
  </port>  
</service>
```

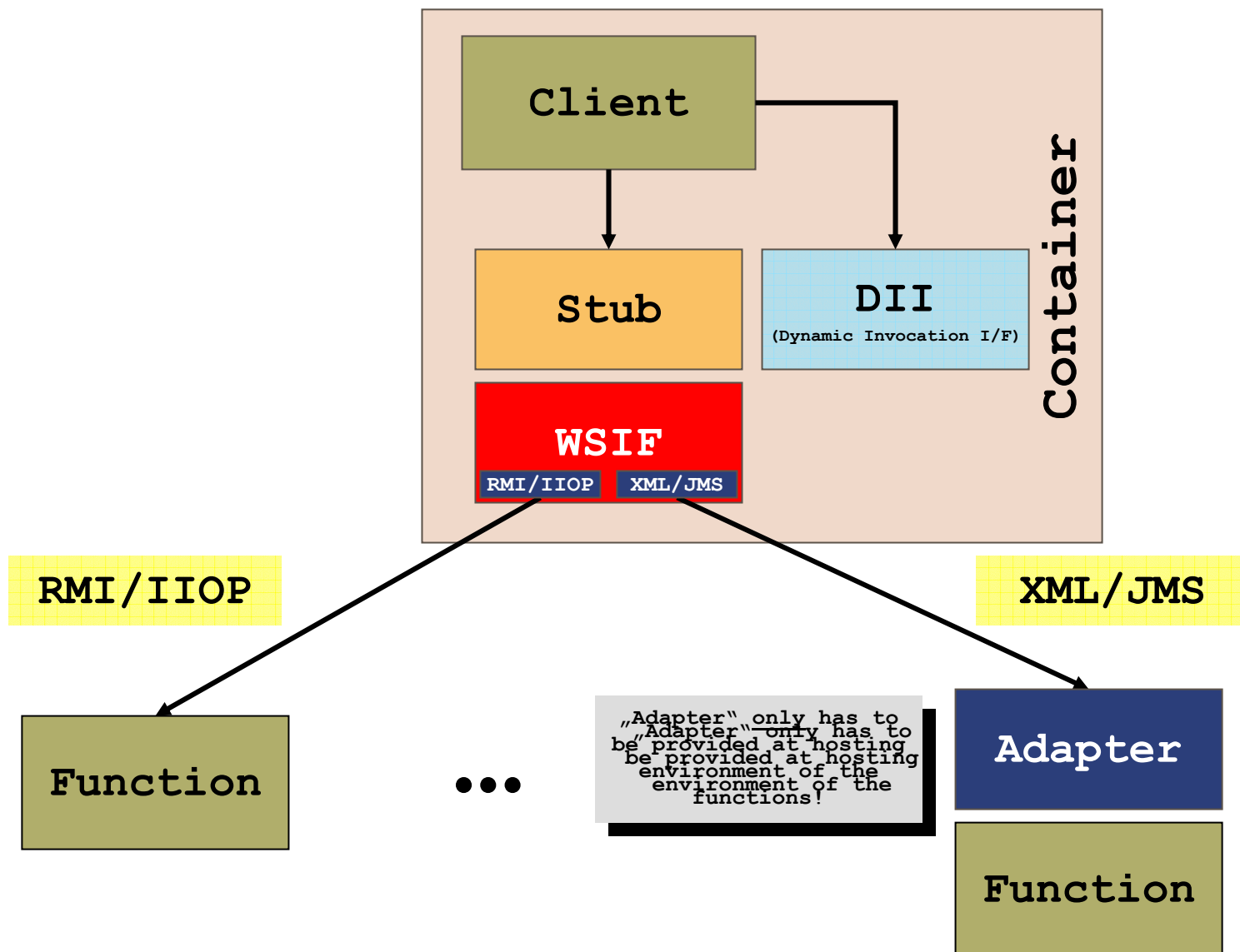
Invoking Web Services

JAX-RPC : Connecting SOAP-Based Web Services In J2EE

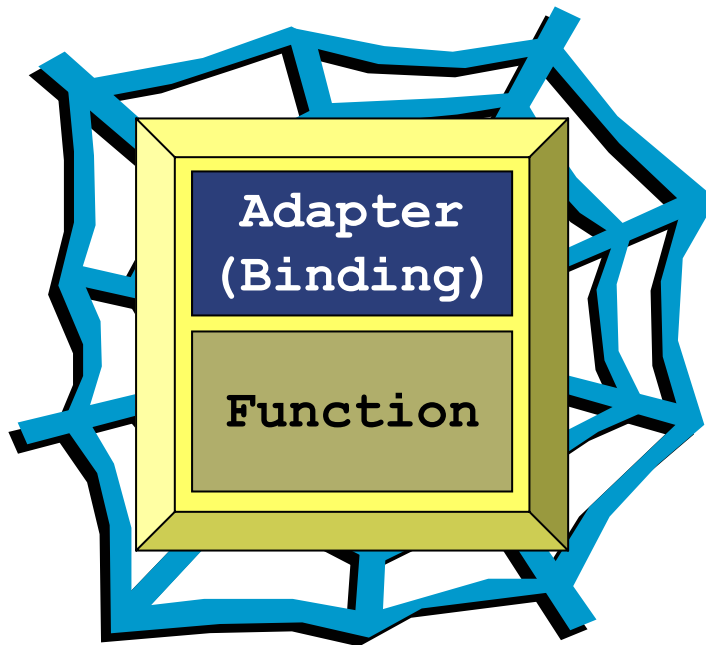


„Adapter“ only has to be provided at hosting environment of the functions!

WSIF: Connecting All Web Services In J2EE

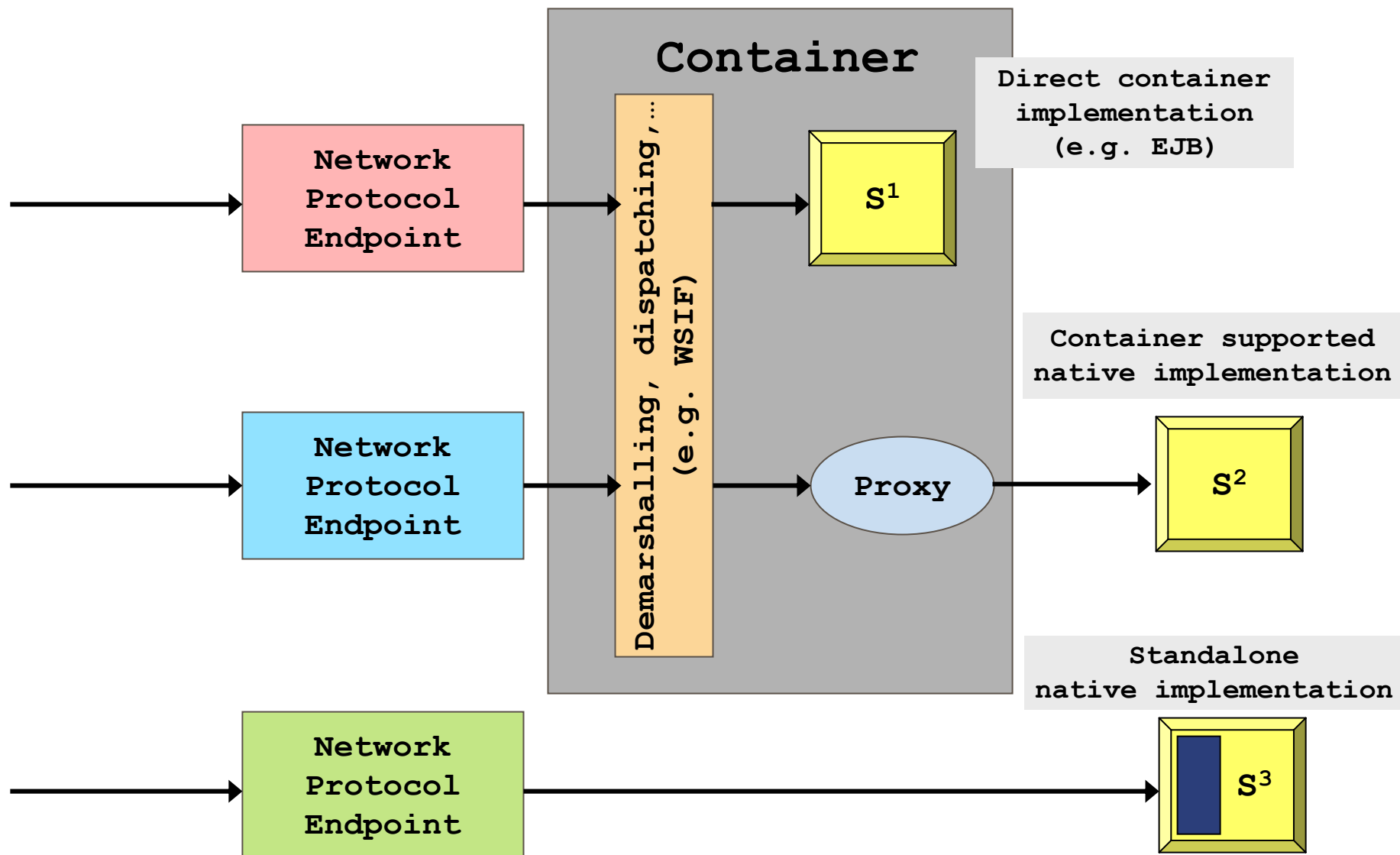


Implementation Of Web Service (Port)

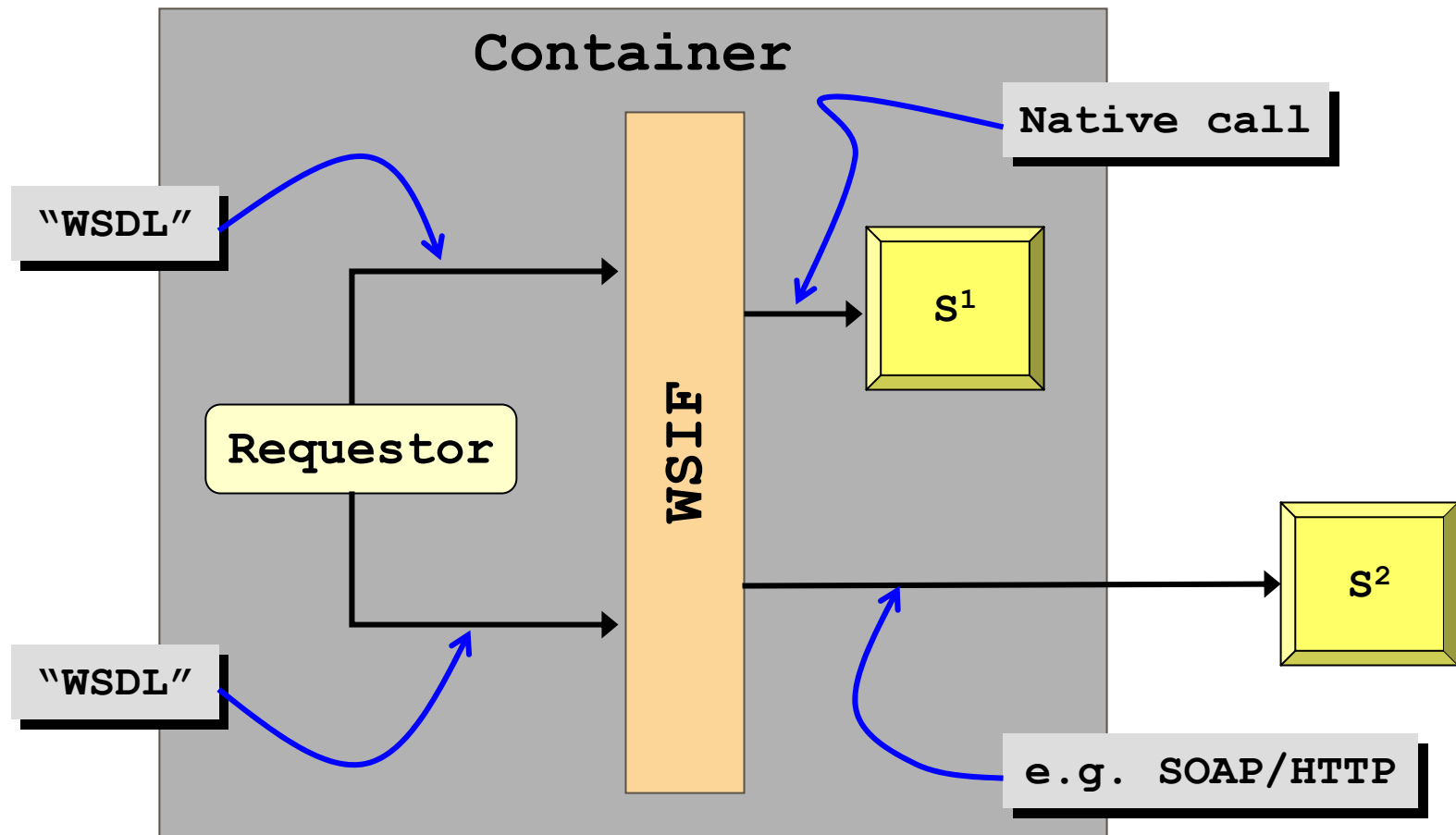


Service providers build adapters for the hosting platforms only, i.e. no separate adapters needed for each client platform.

Role Of Containers (Integrating WSIF)



Invoking Services Within A Container

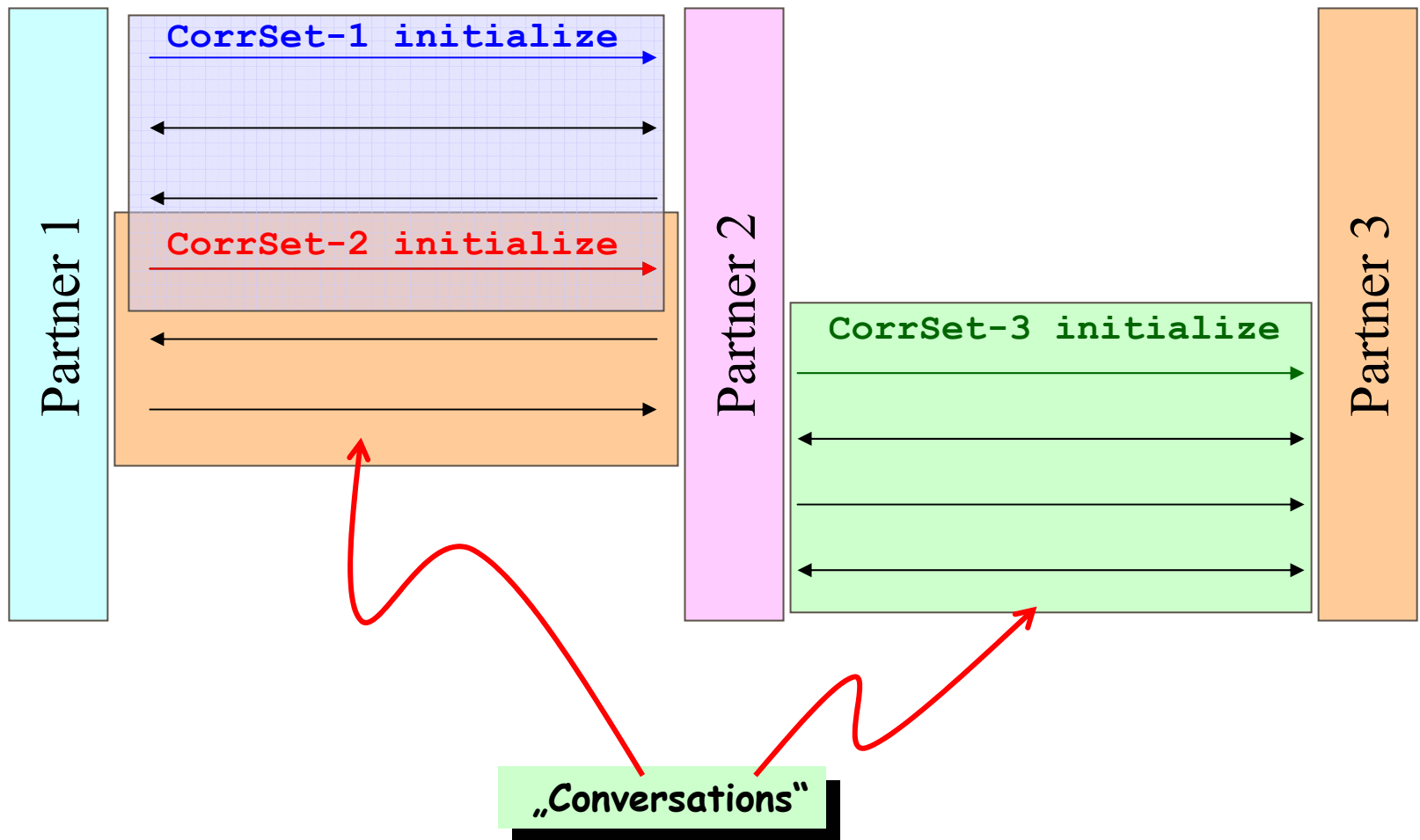


Lifecycle Of Web Services

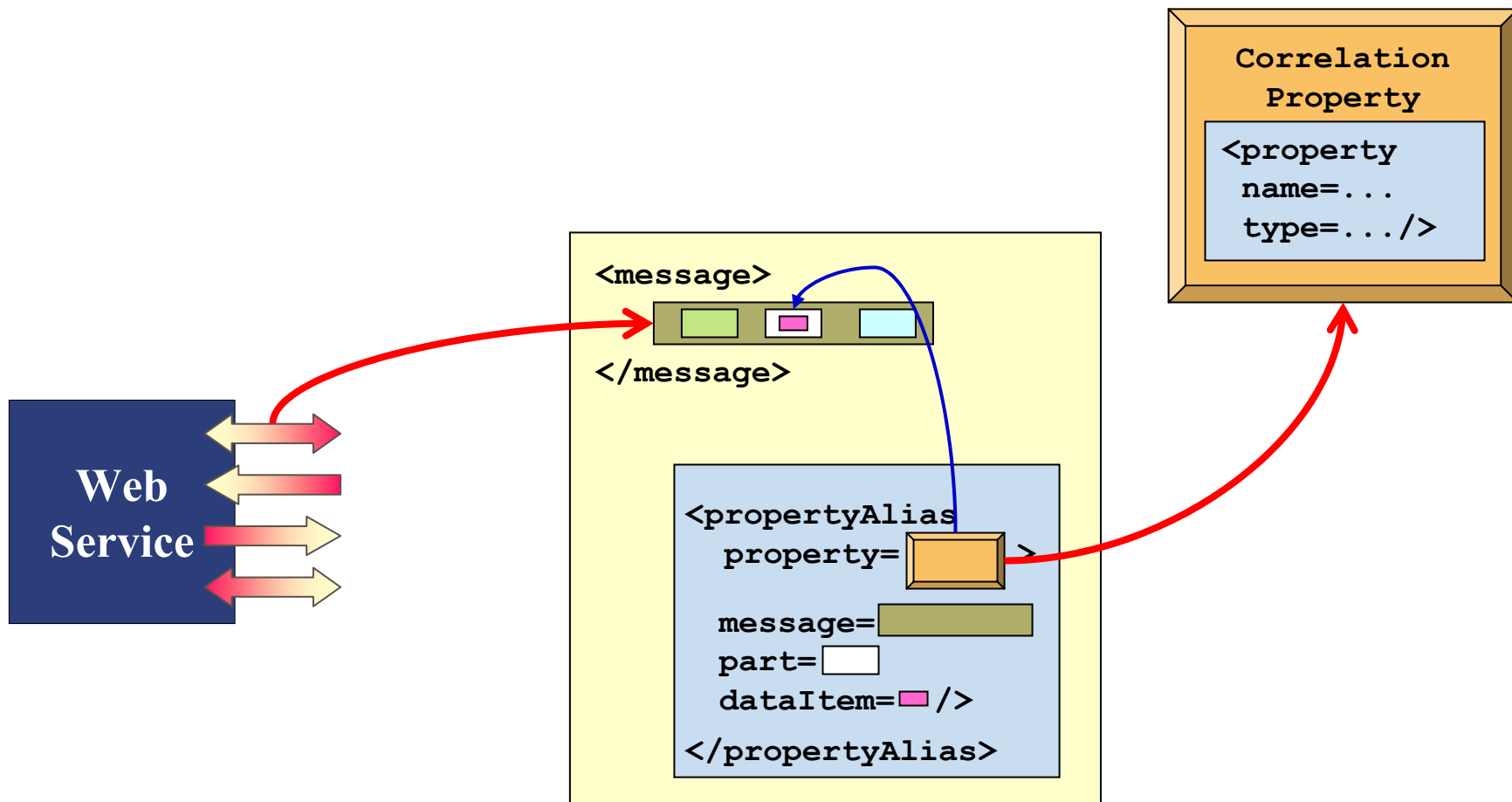
State And Web Services

- Web service can be stateful
 - Based on ability to support conversations, i.e. long-running interactions like business processes
 - Based on being an entity, i.e. having “identity”
- Implicit state management, e.g. BPEL
 - Correlation properties embedded in application messages exchanged
 - New set of correlation properties implicitly creates new “conversation” (= process instance in case of BPEL)
- Explicit state management, e.g. OGSA
 - Resources have observable state and identity
 - Lifecycle functions to manage each resource
 - But: Softstate management – see later

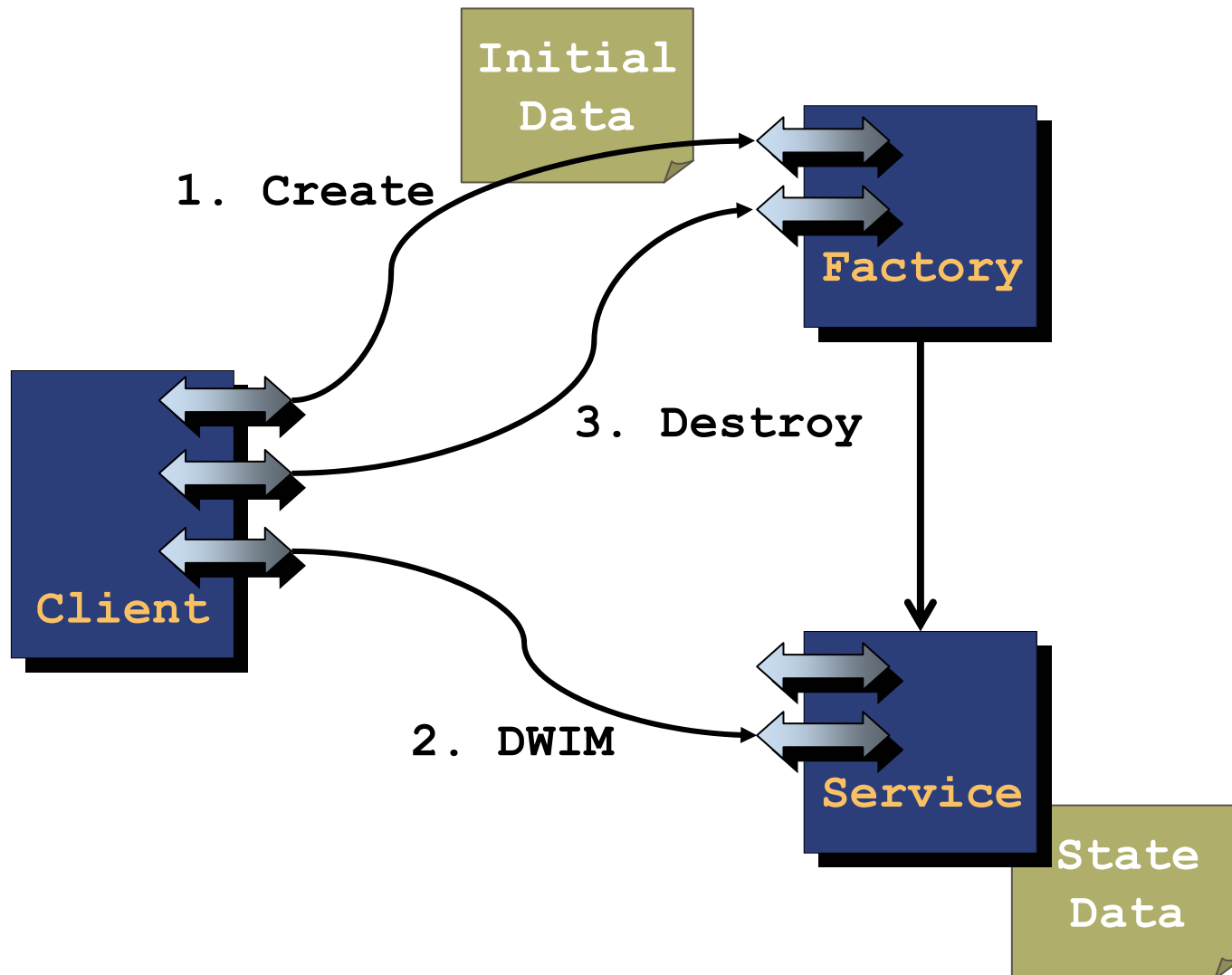
Conversations: Implicit State



Correlation Properties: The Mechanics



Factory: Explicit State

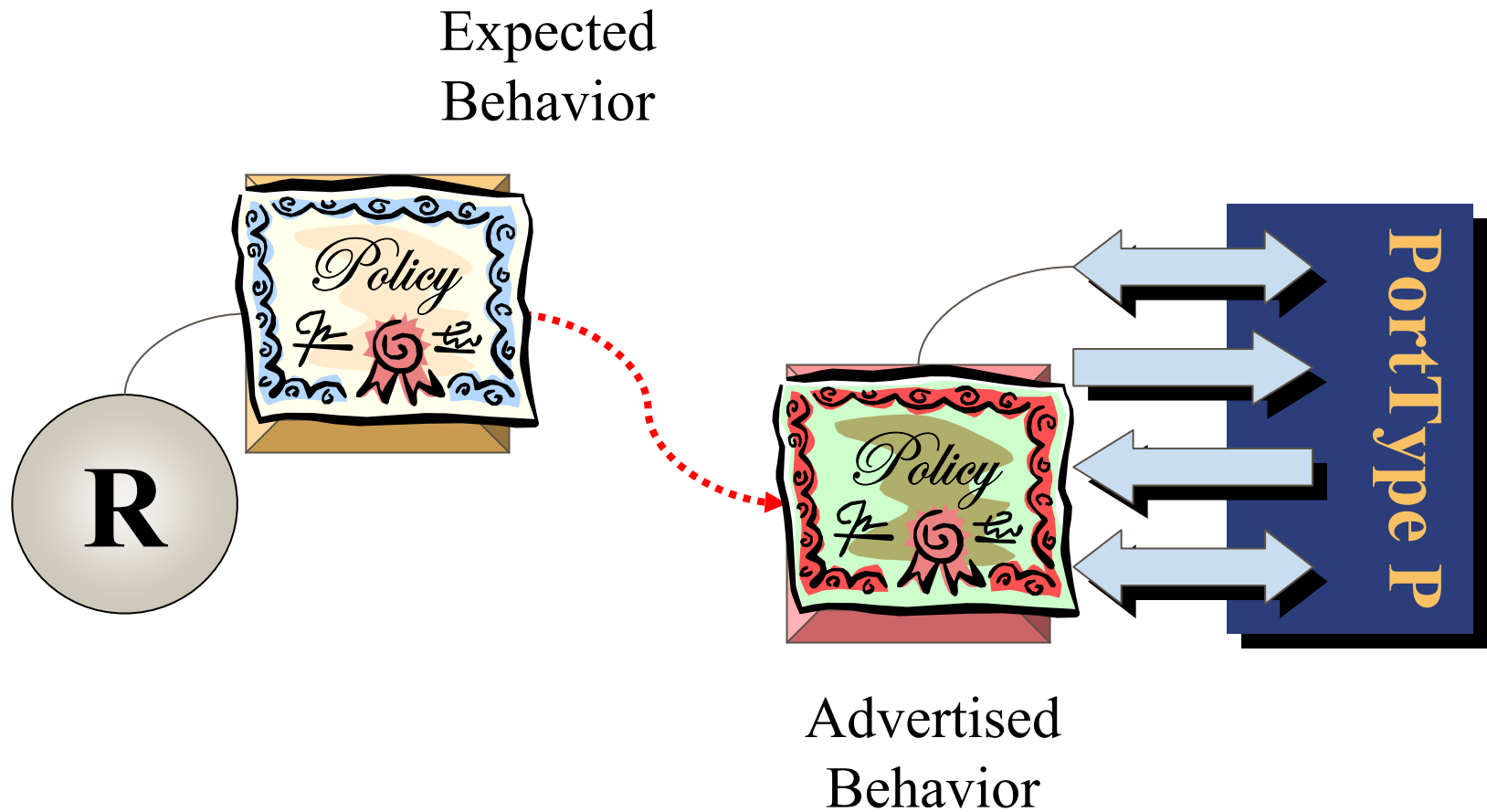


Client Programming Model

- The client is impacted by the fact whether state is transparent or opaque
- Explicit state management
 - Client must know that/how newly discovered service has to be explicitly managed wrt lifecycle
 - Inconvenient in dynamic environments
- Implicit state management
 - Client doesn't know whether or not service is stateful or not
 - Very convenient in dynamic environments

Policies

Quality Of Services & Policies



Policy: Example

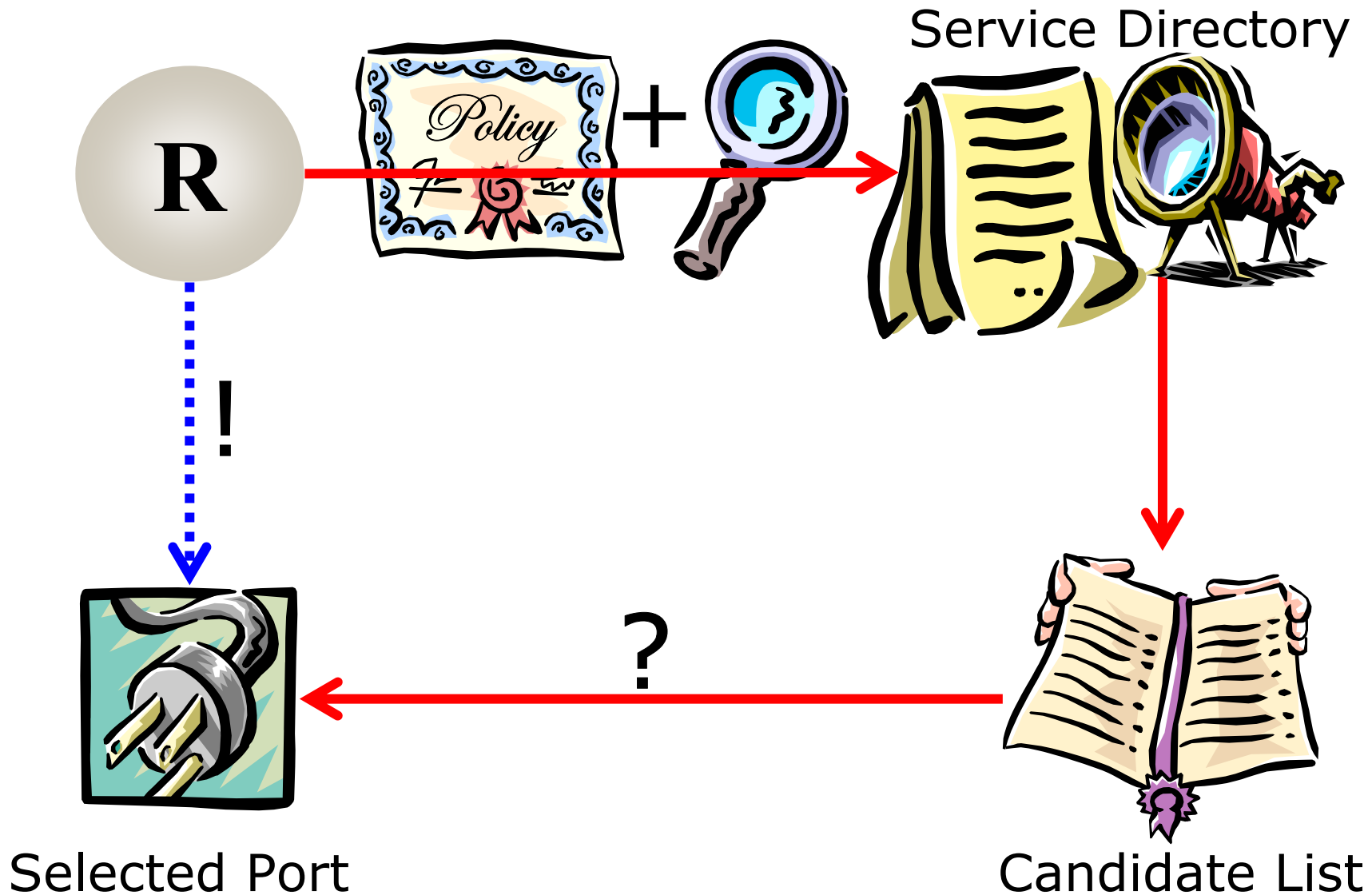
```
<wsp:Policy xmlns:SecurityNS="..." xmlns:my="...">
  <wsp:ExactlyOne>

    <my:onlinePayment>
      <wsp:ExactlyOne wsu:Id="opts">
        <my:creditCard wsp:Usage="wsp:Required" />
        <my:cheque wsp:Usage="wsp:Required" />
        <my:transfer wsp:Usage="wsp:Required" />
      </wsp:ExactlyOne >
      <my:sessionStart Usage="wsp:Required" />
    </my:onlinePayment>

    <my:subscriptionPayment>
      <wsp:PolicyReference URI="#opts"/>
      <wsp:monthly Usage="wsp:Required" />
    </my:subscriptionPayment>

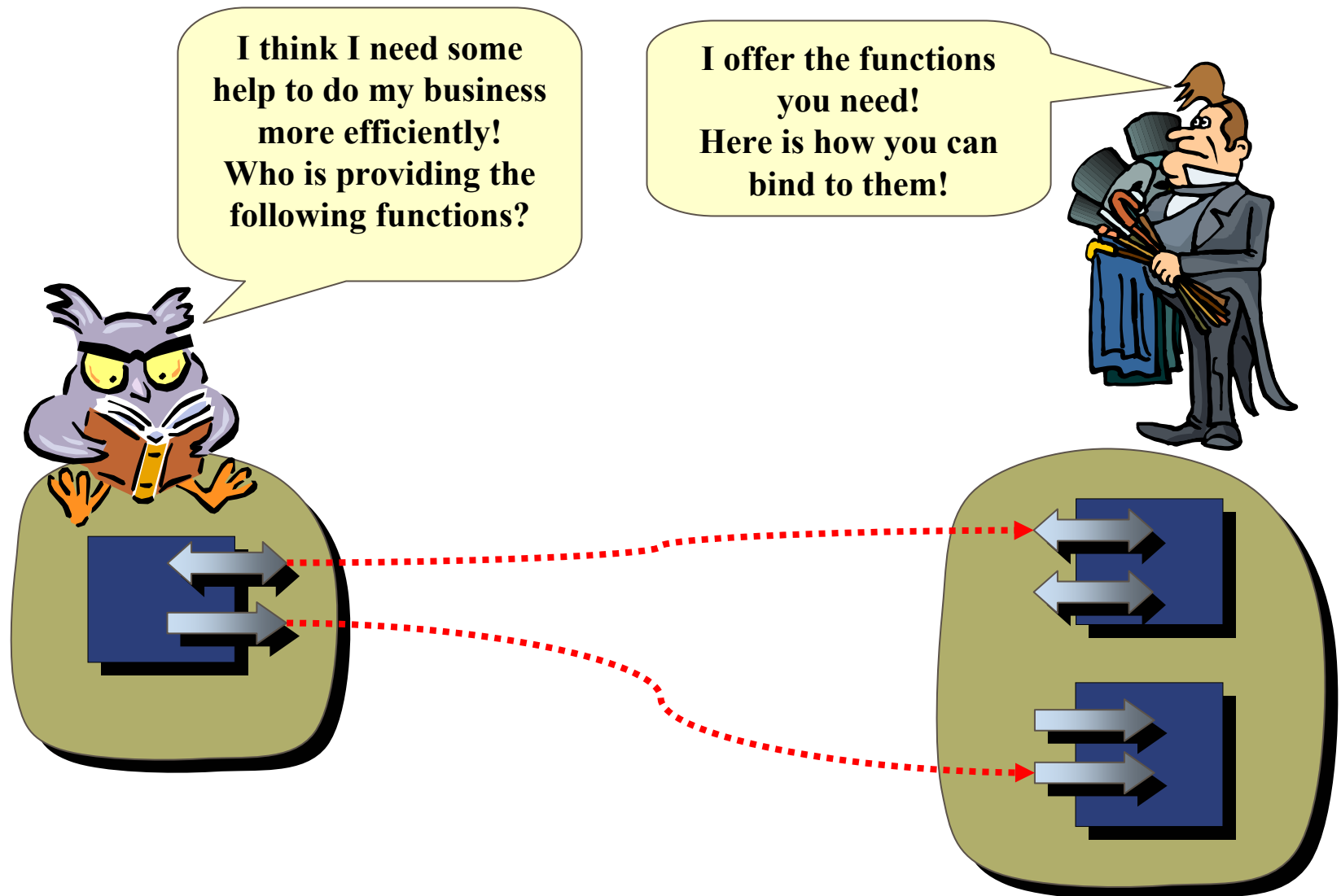
  </wsp:ExactlyOne>
</wsp:Policy>
```

Matching Endpoints Based On Policies

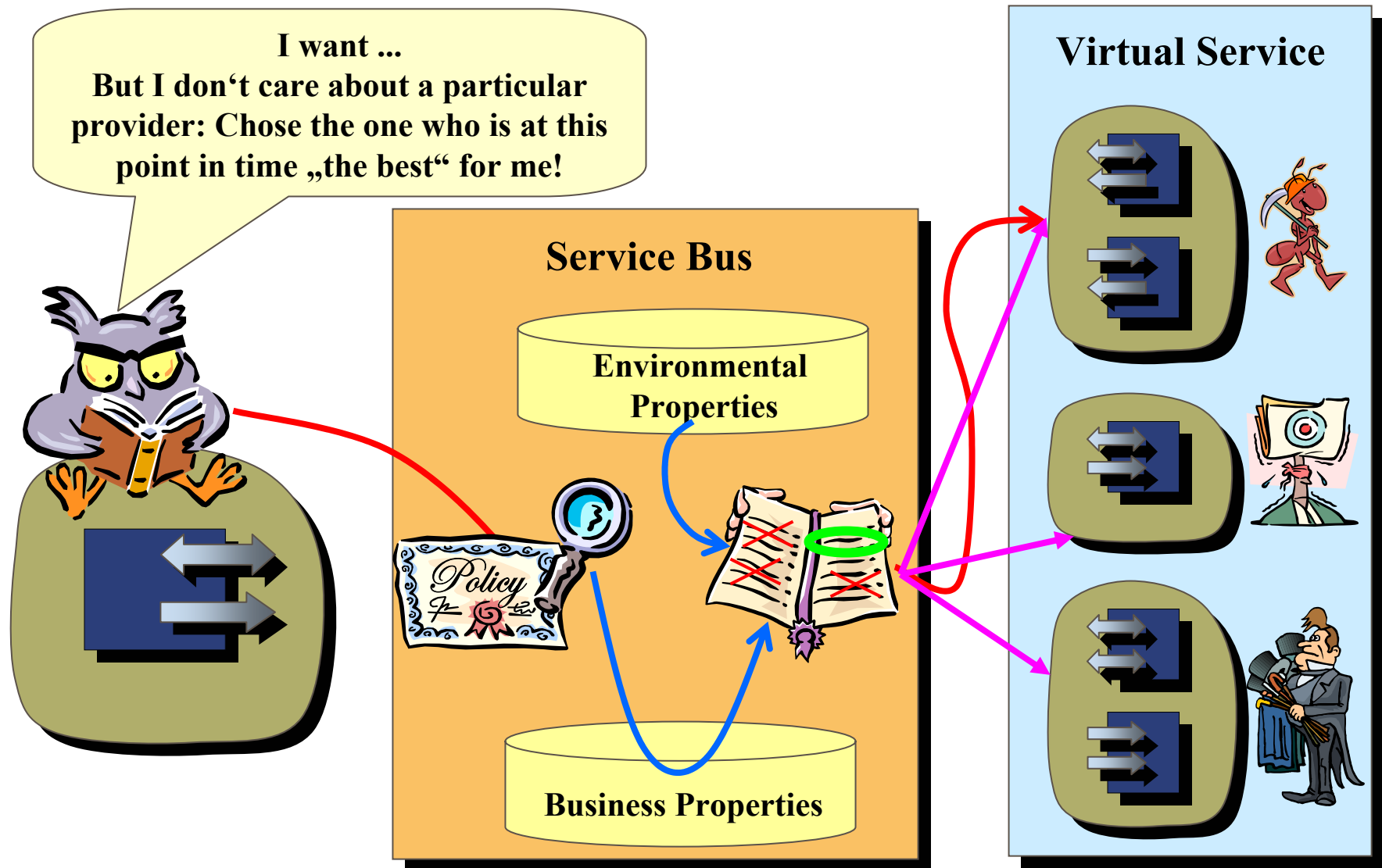


Service Bus

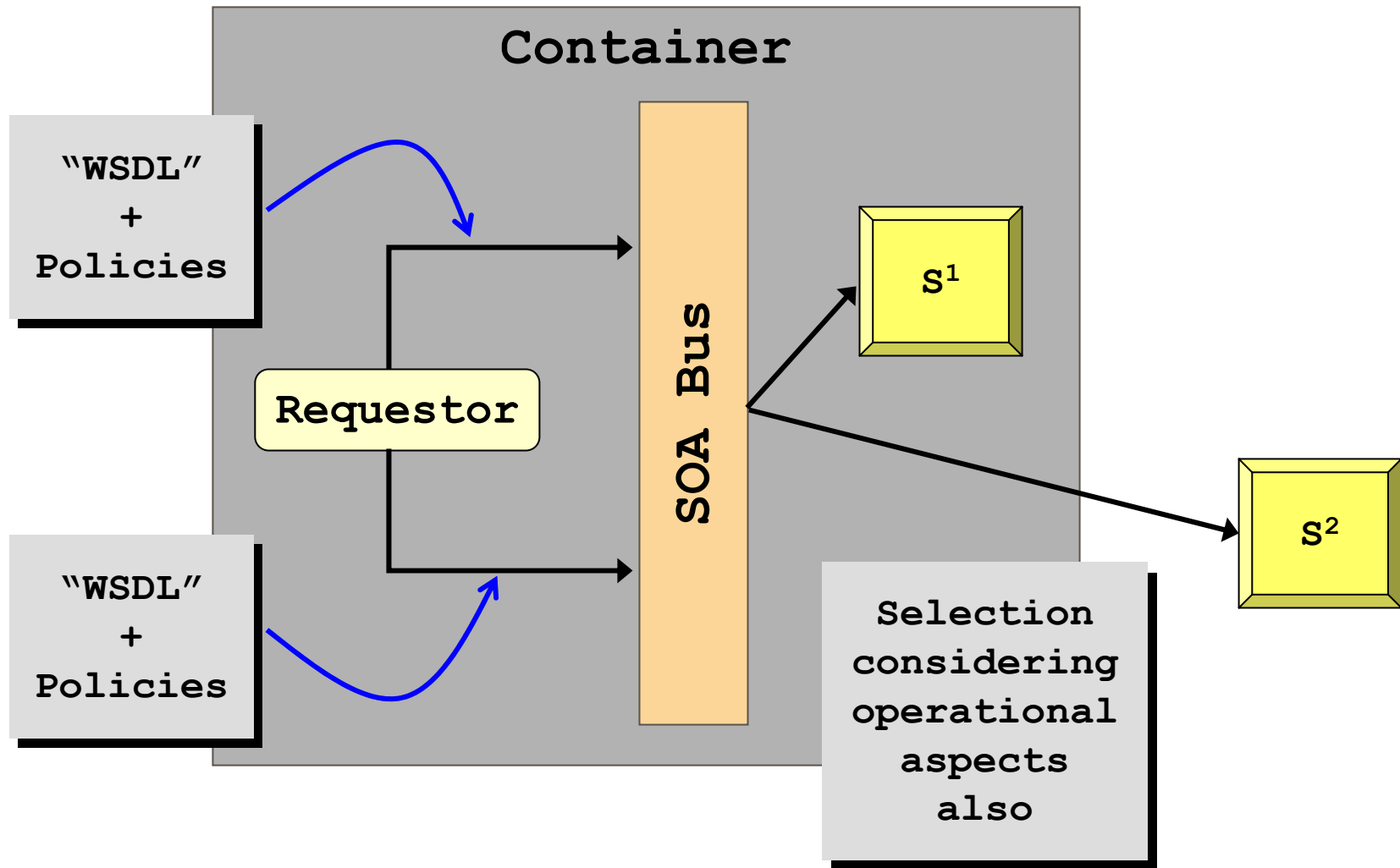
Dynamic Selection Of Services



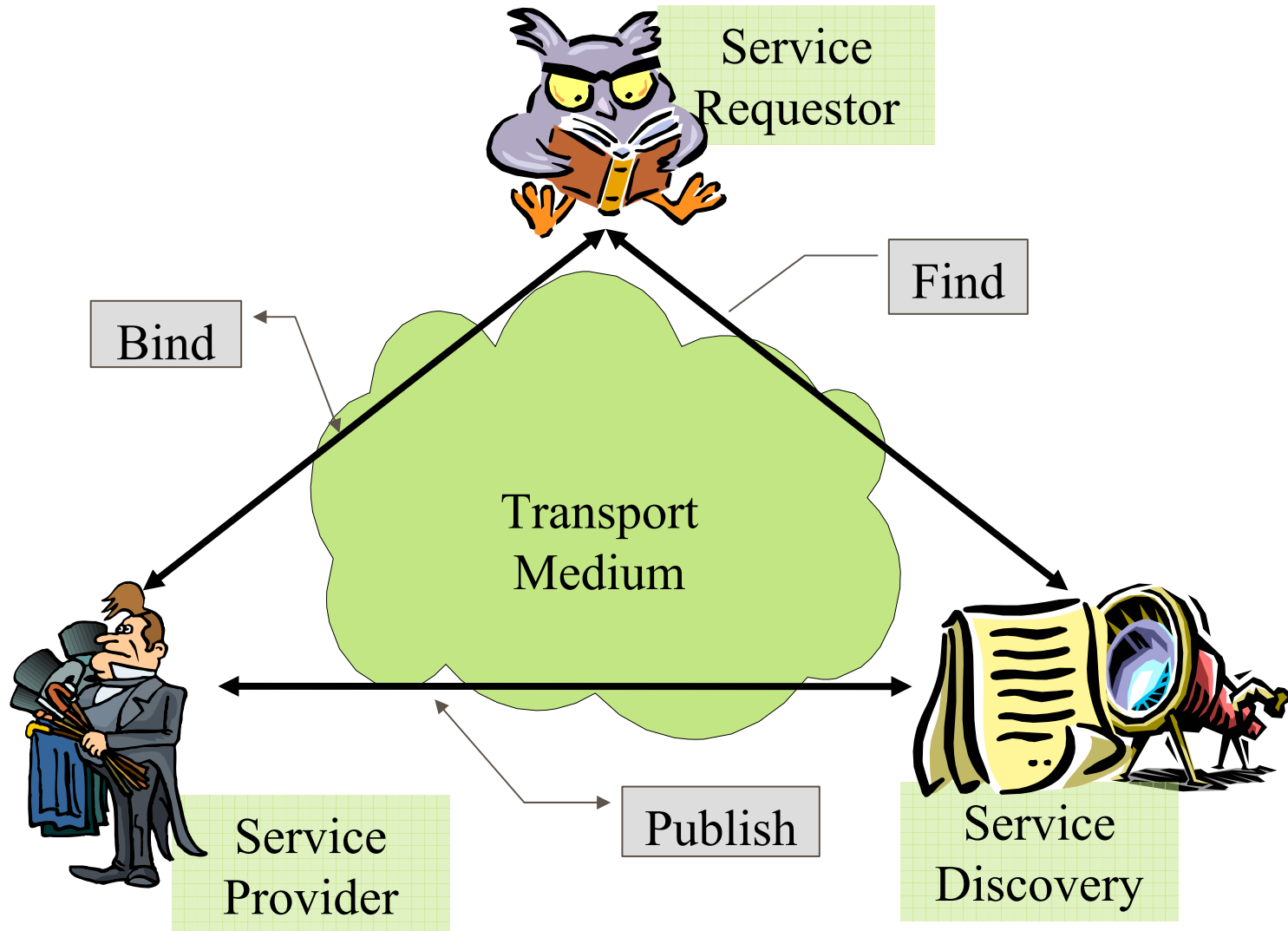
Don't Care! Virtualizing Services



Virtualization And Container



Service Oriented Architecture (SOA)



Sample Work To-Be-Done...

To Dos

- Define and discuss component models for Web services – pros and cons
- Stateful and stateless Web services
 - How to hide differences for clients
 - Impact on service bus
- Policies
 - Efficient combination/aggregation of policies into single policy documents
 - Dynamic matchmaking of policies

Agenda

Introduction

Virtual Components

Virtual Environments

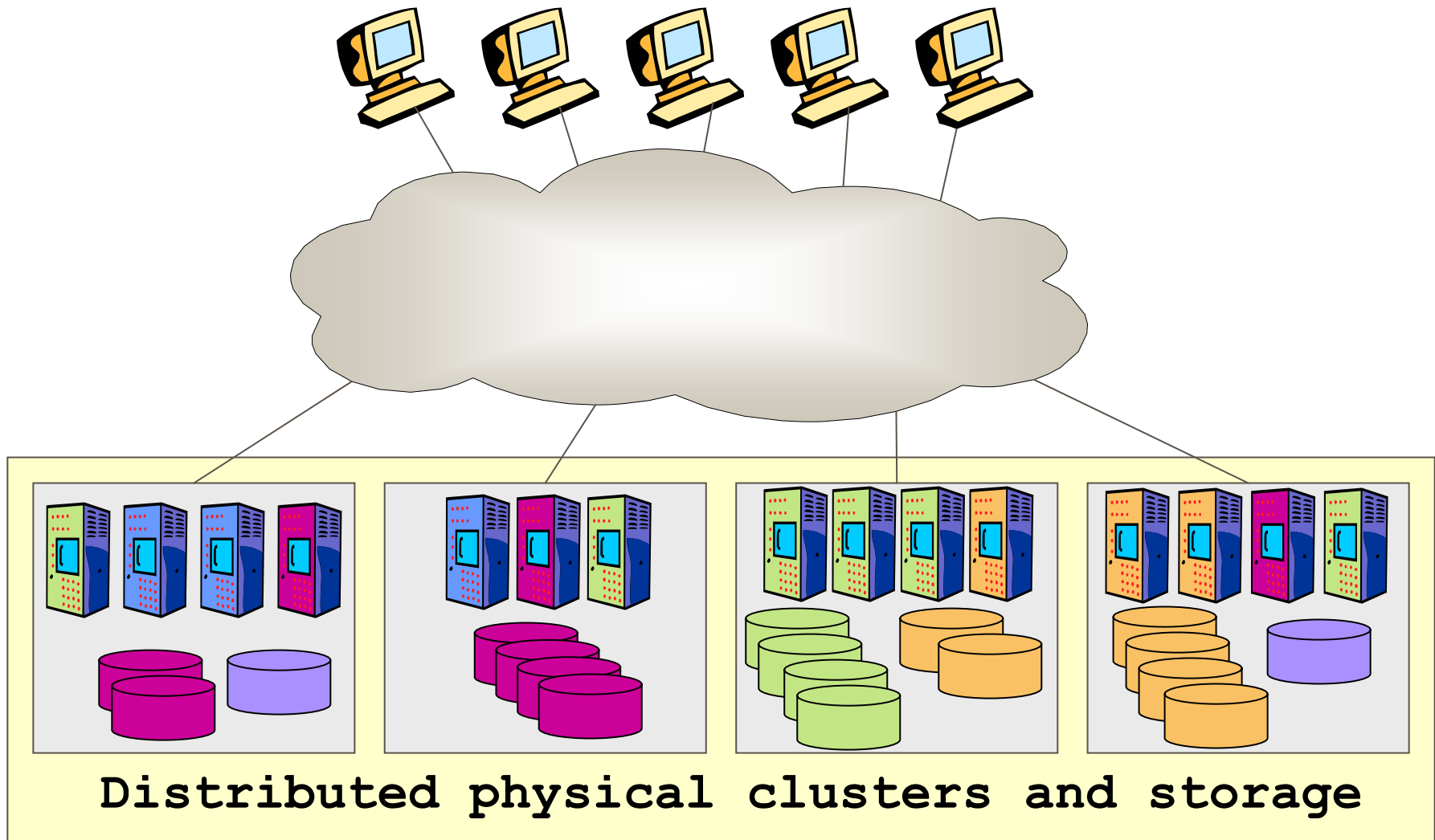
Application Structure

Aggregations

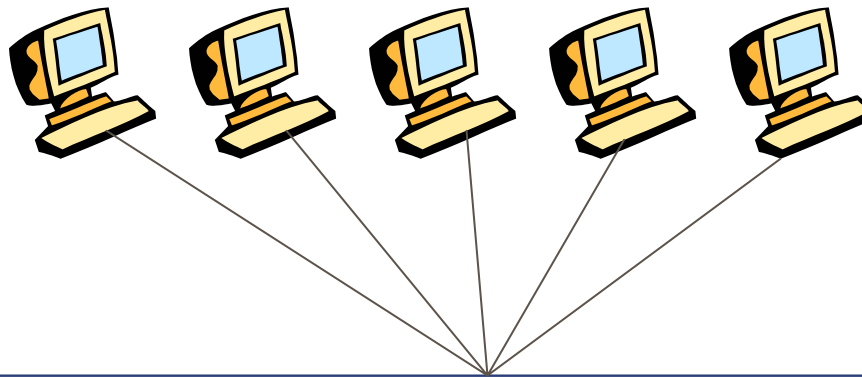
Summary & Conclusion

Virtualizing Compute Resources

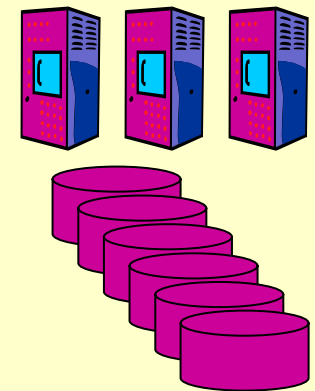
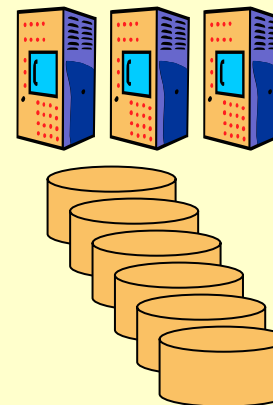
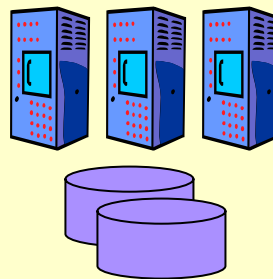
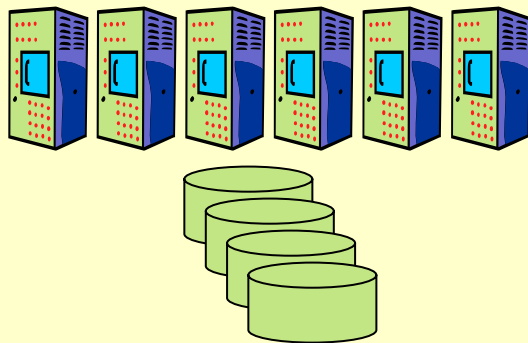
Distributed Heterogeneous Environment



The Grid: Virtualizing Resources

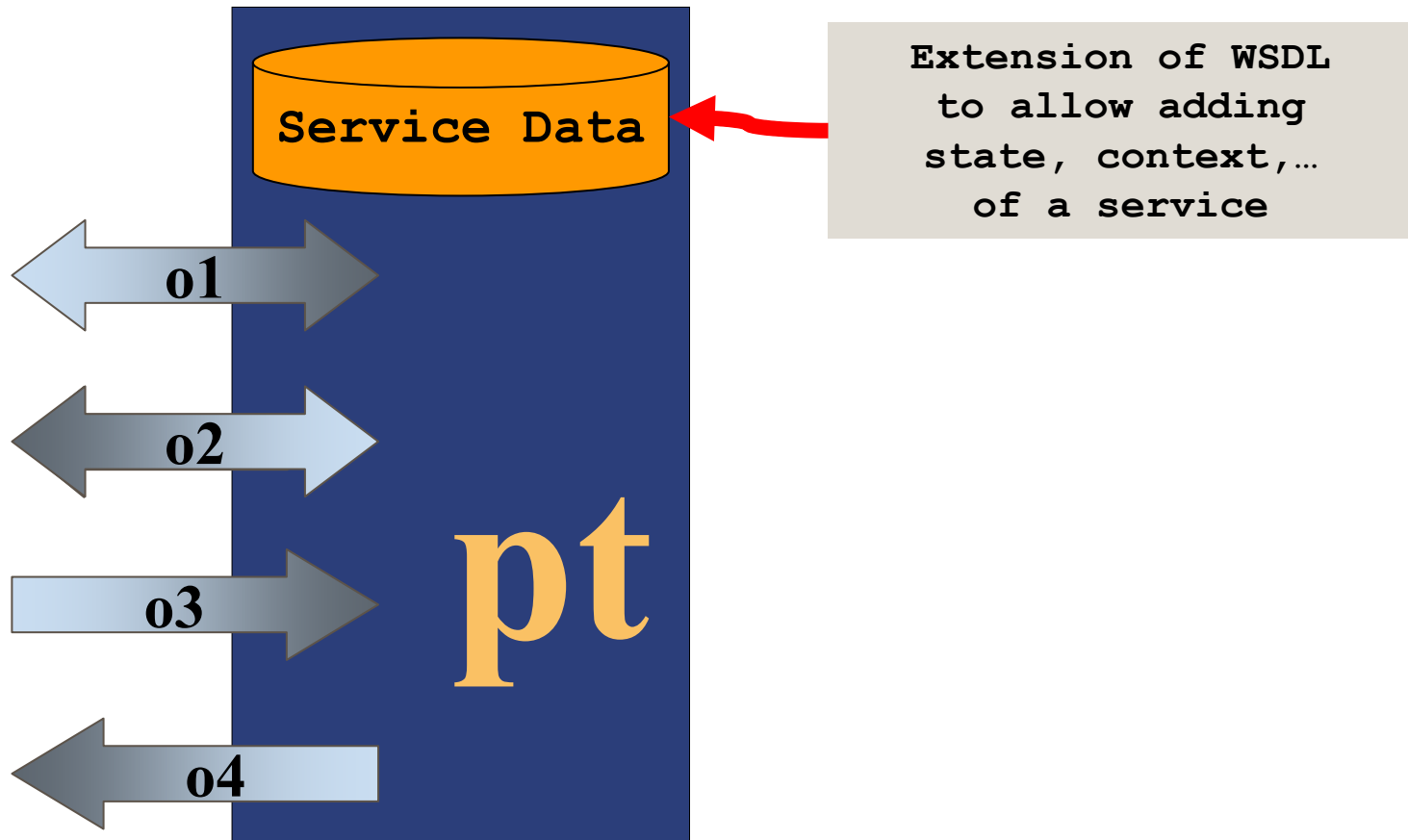


Grid Middleware



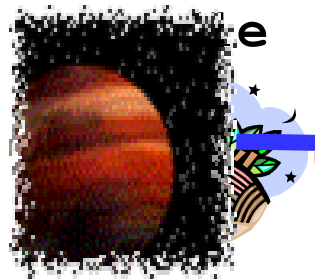
Virtual clusters and storage

Grid Service Description

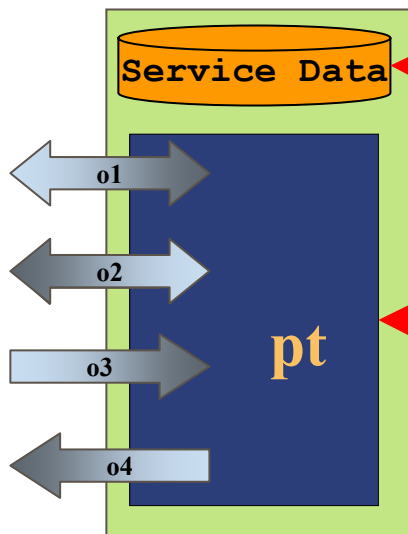
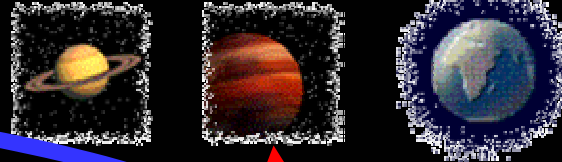


Grid Service Instance: Picture

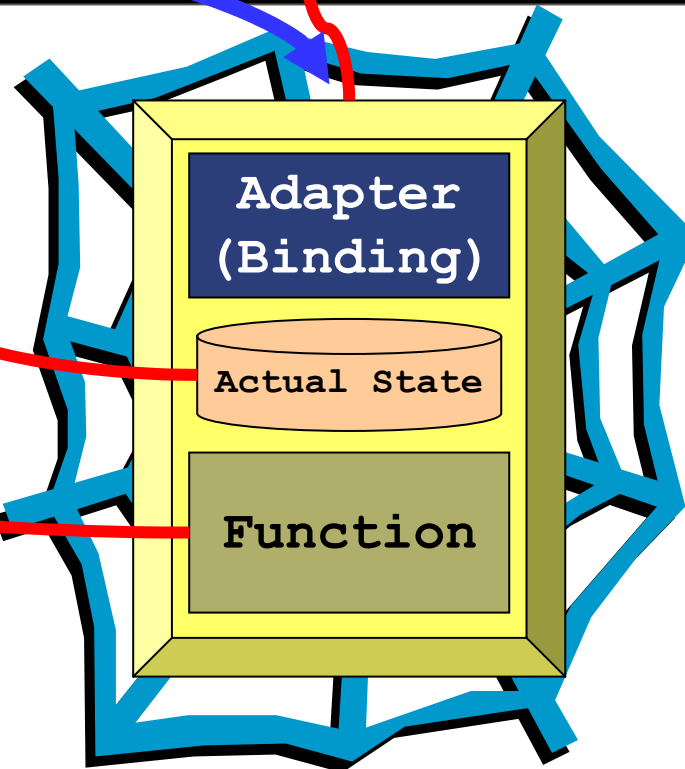
Grid Service



Grid Service References

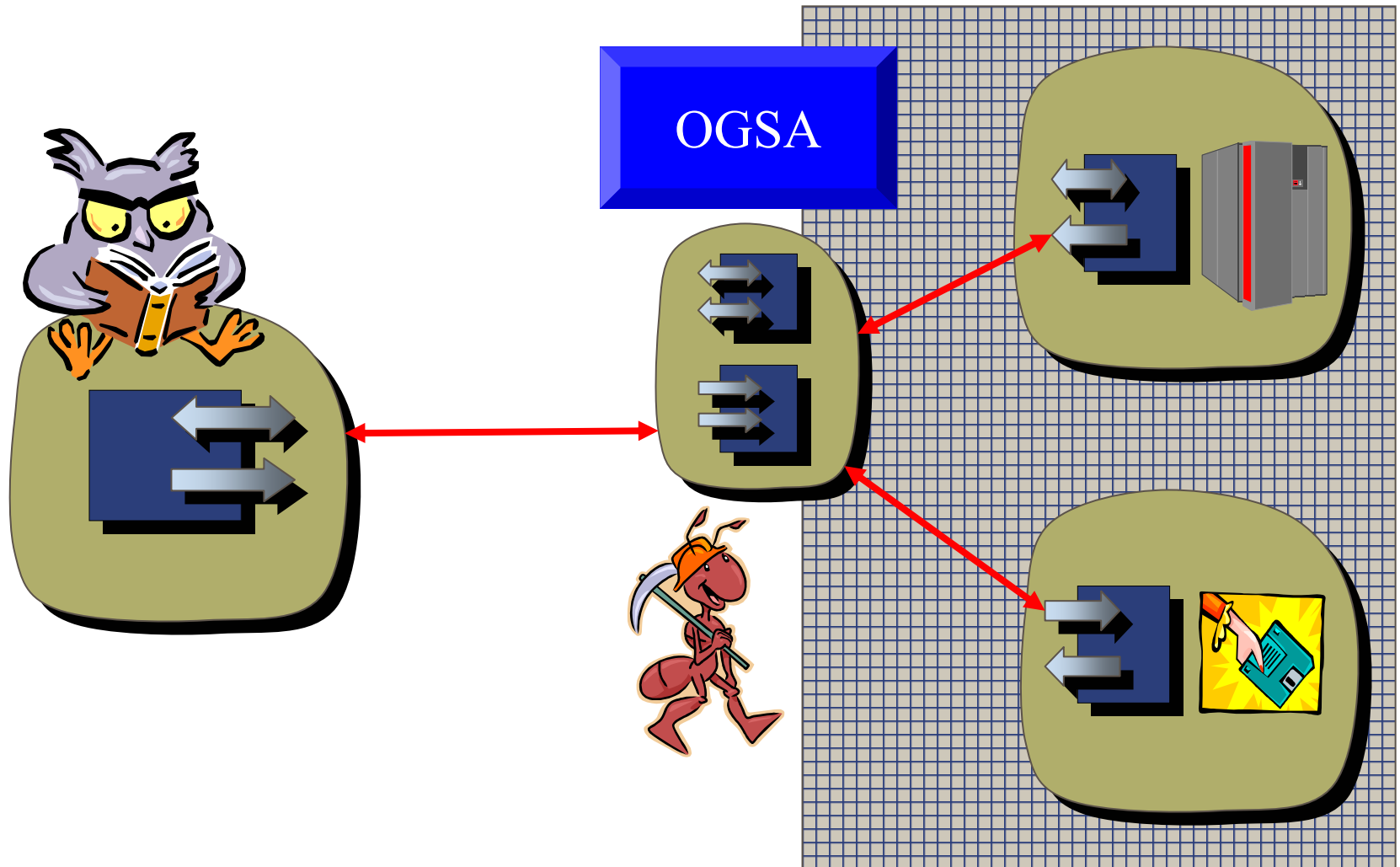


Grid Service
Description

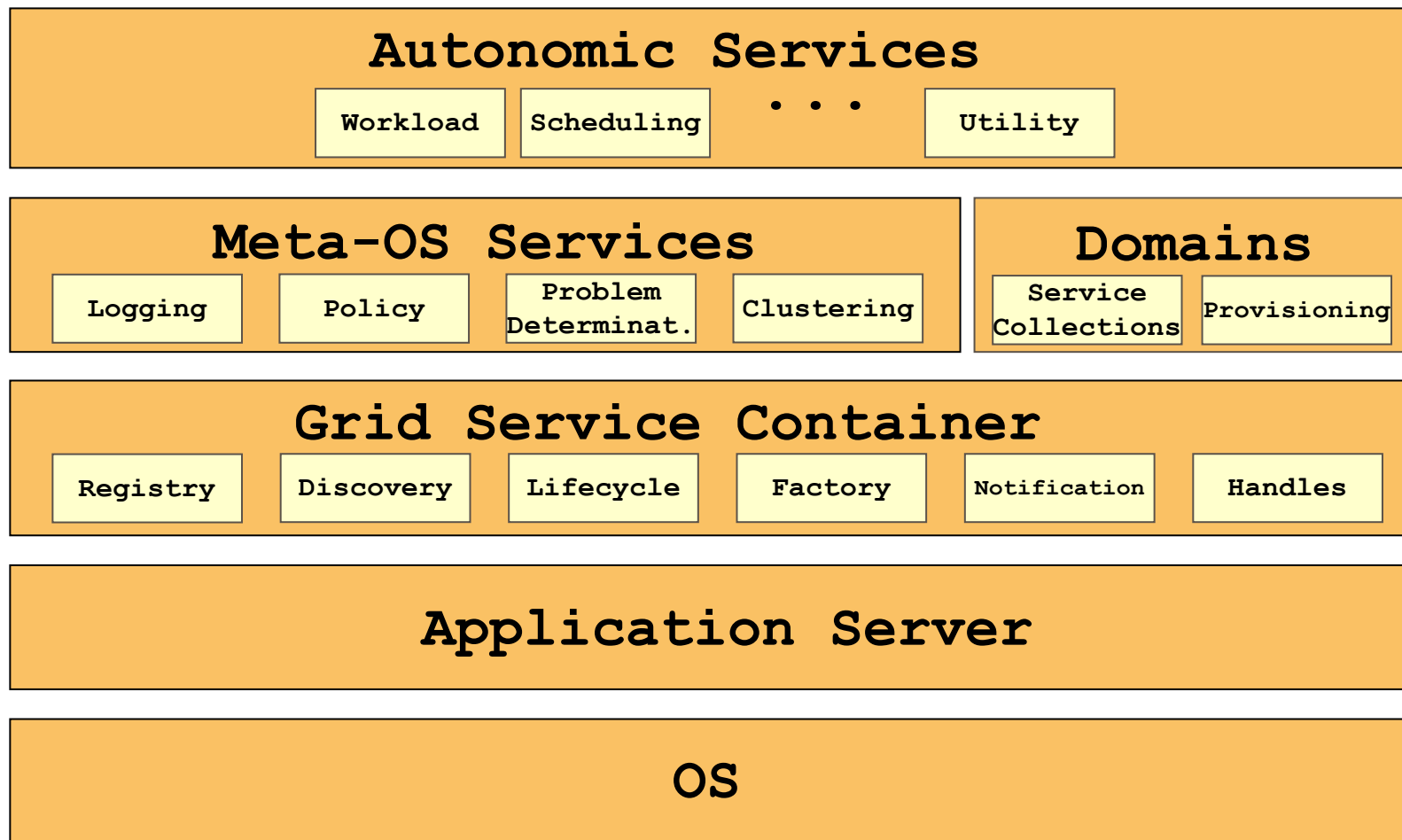


Grid Service Instance

Infrastructure Grid

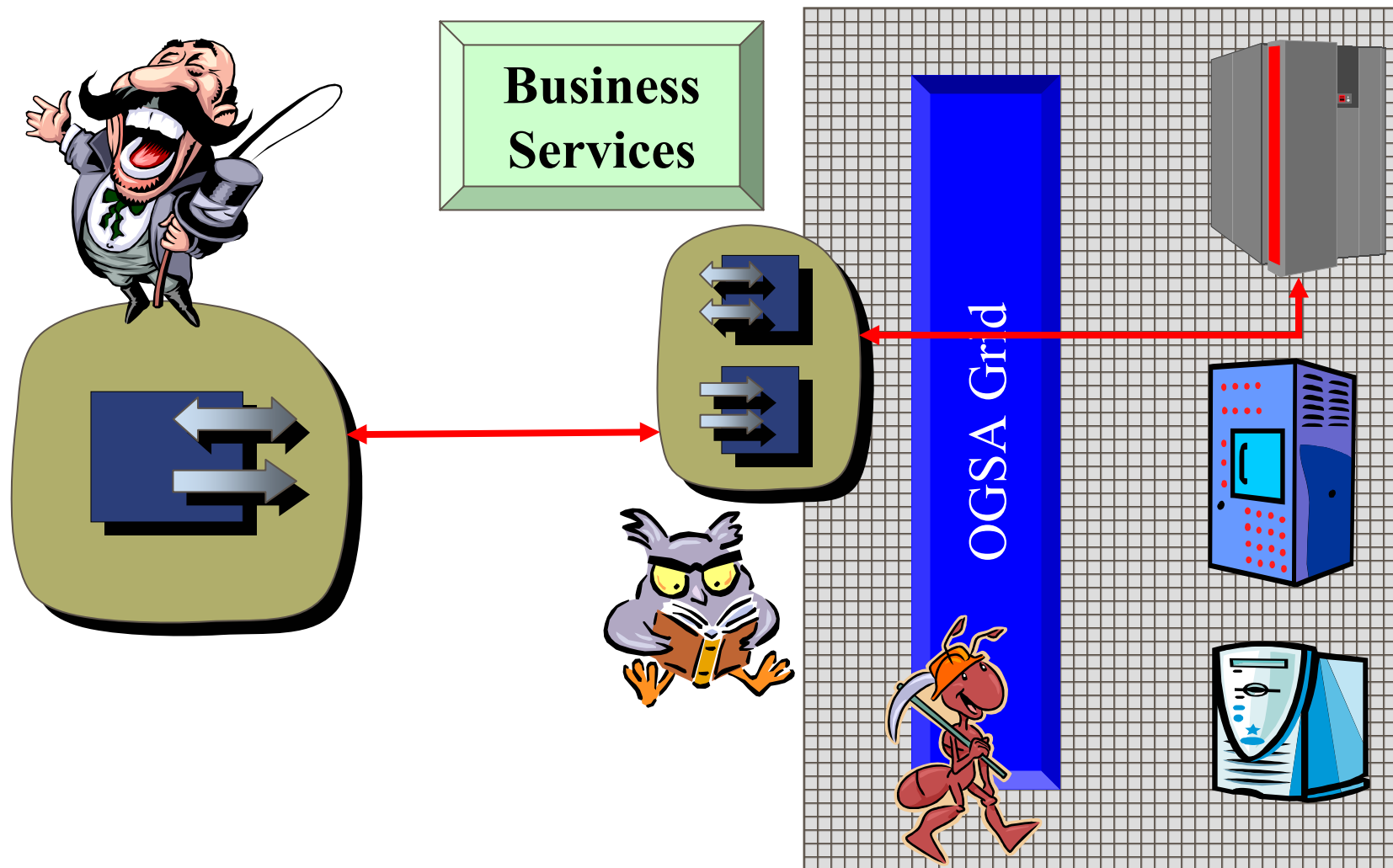


Overall Grid Services Stack

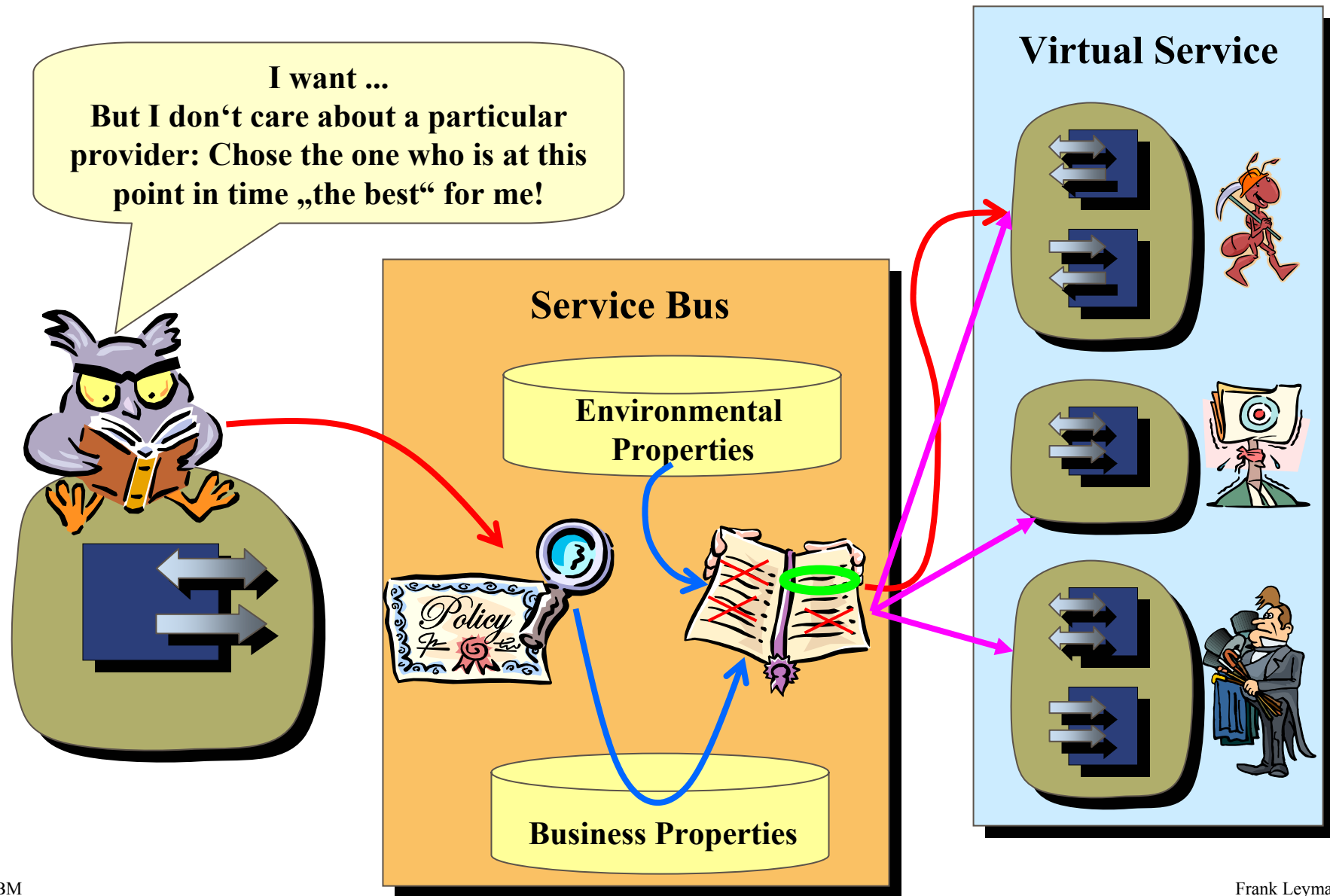


“On Demand”: Sunrise

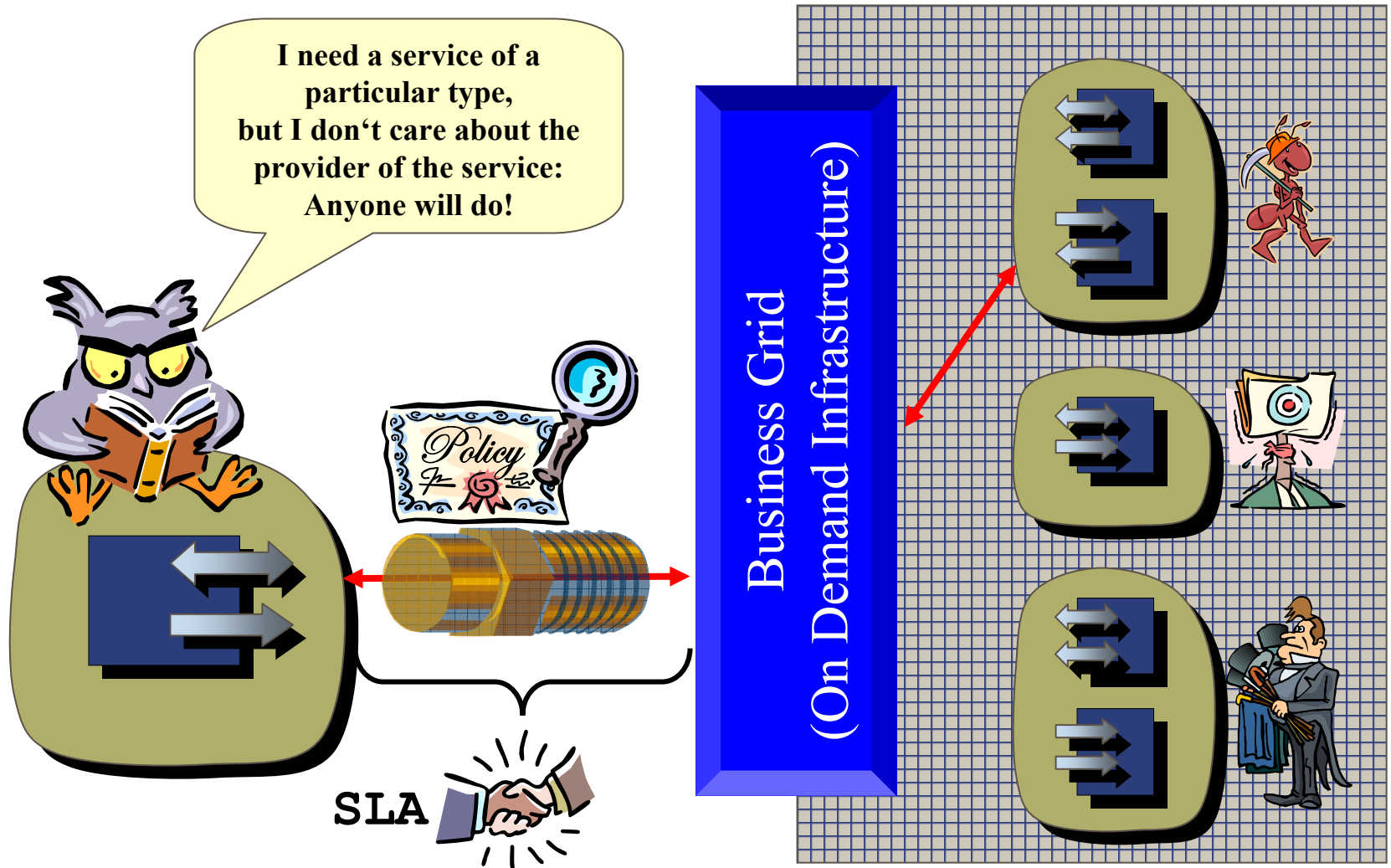
Grid For Building Business Web Services



Reminder: Virtualizing Services



Business Grid



Sample Work To-Be-Done...

To Dos

- Architecture and design for a Grid service container
 - Interfaces/contracts between implementation of a Grid service and application server for portability
- Meta-Schedulers
- Transitive payment, QoS,...
- Do we need the difference between Grid services and Web services?

Agenda

Introduction

Virtual Components

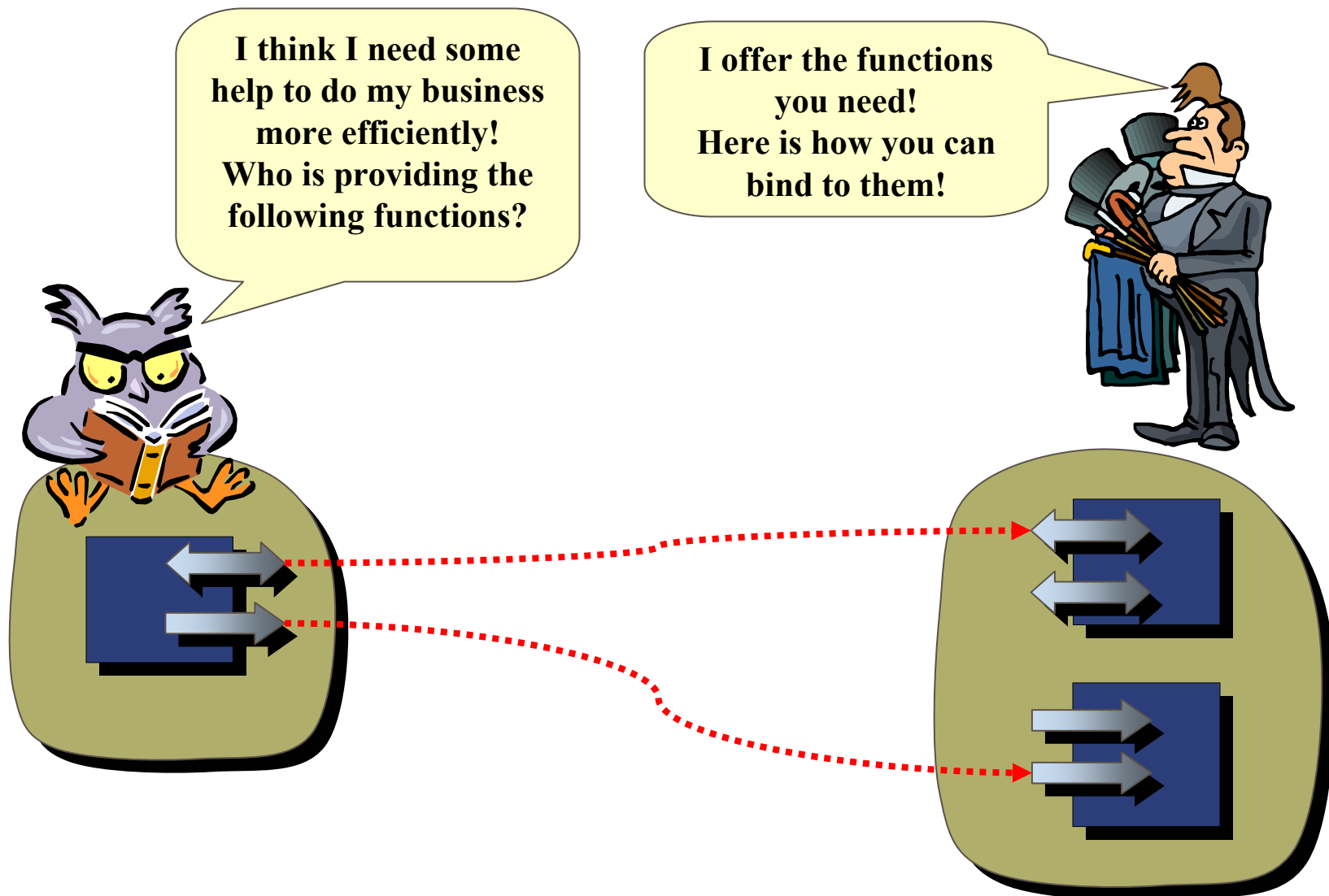
Virtual Environments

Application Structure

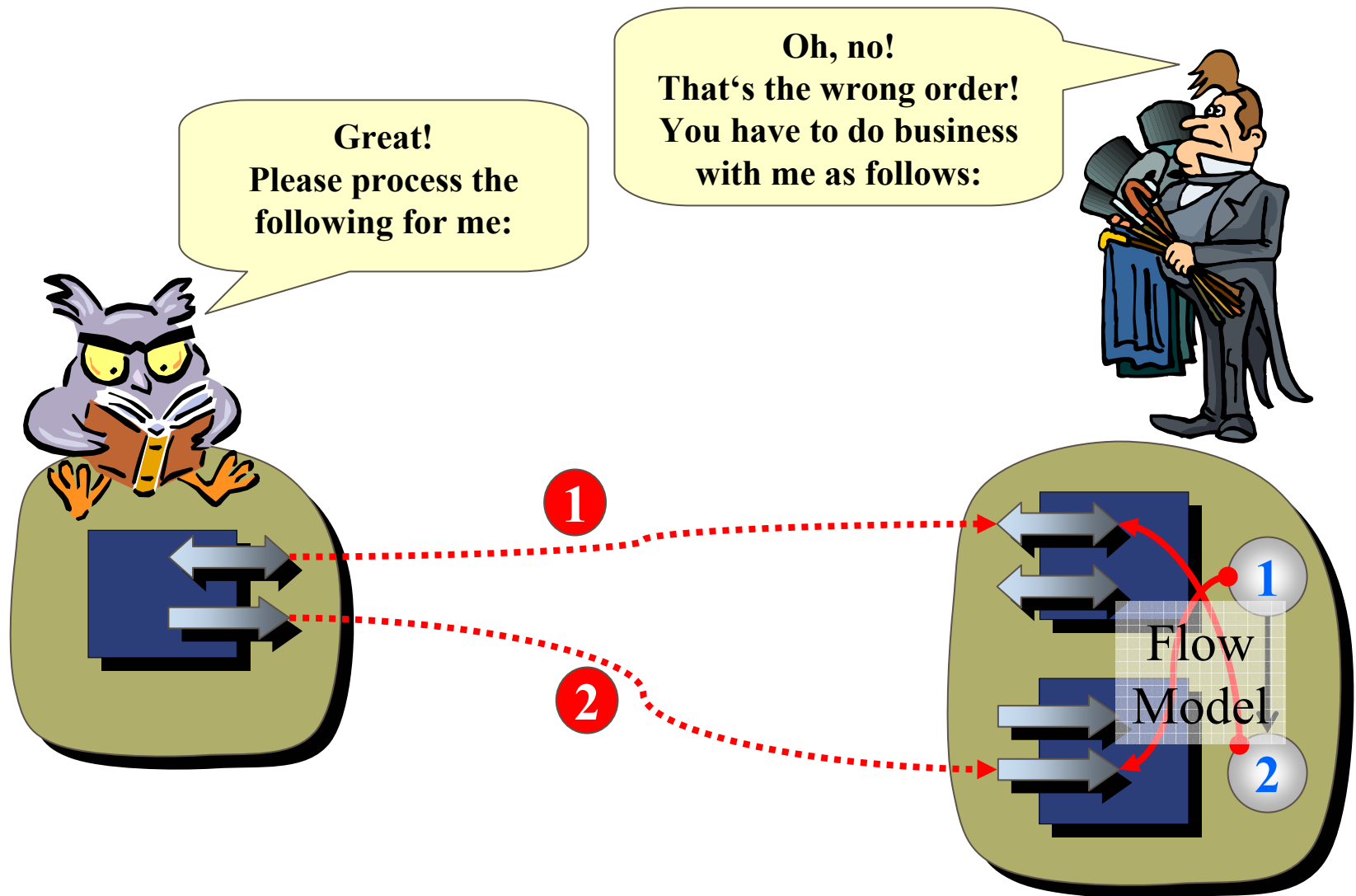
Aggregations

Summary & Conclusion

Reminder ☺

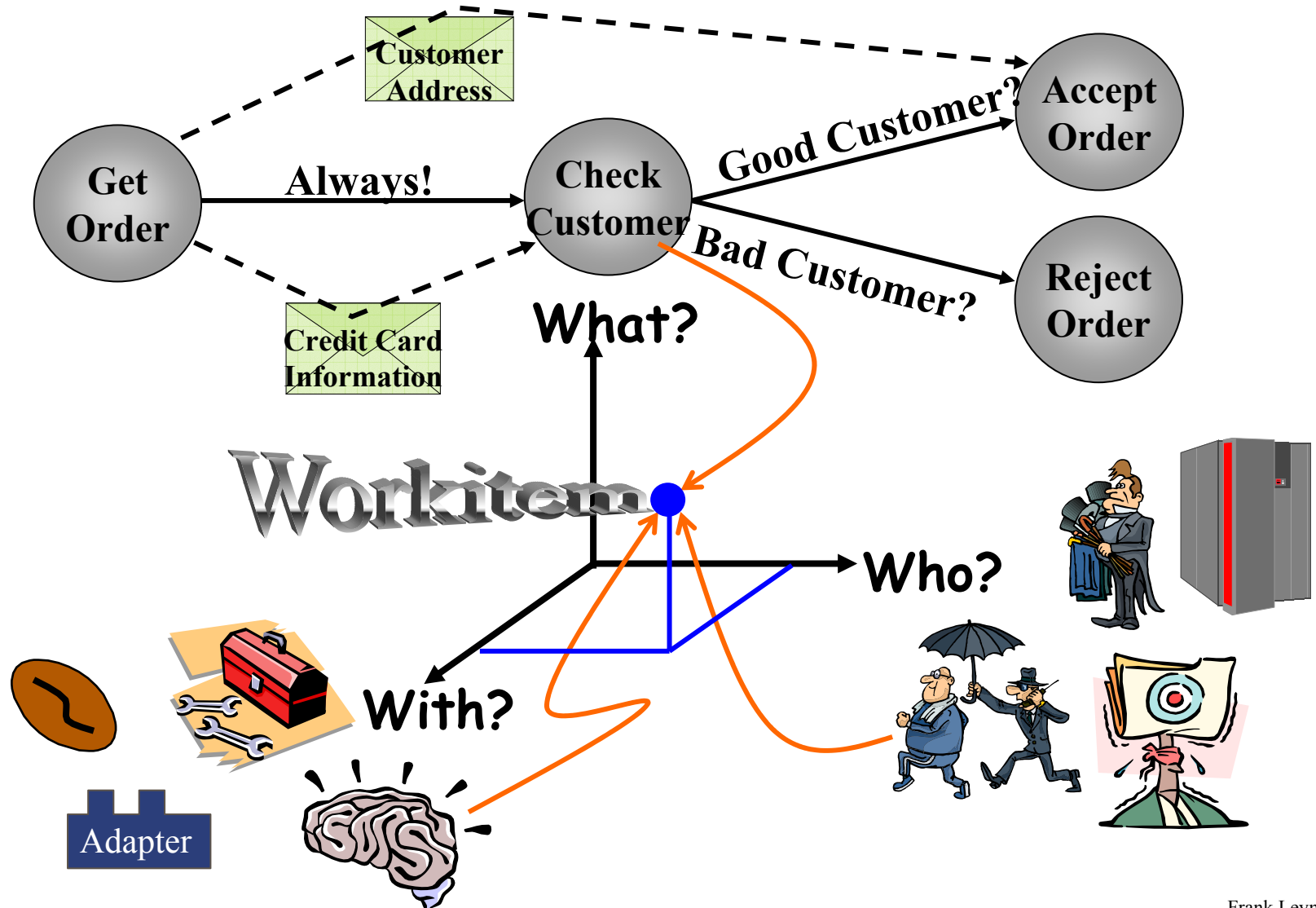


Order Really Matters!

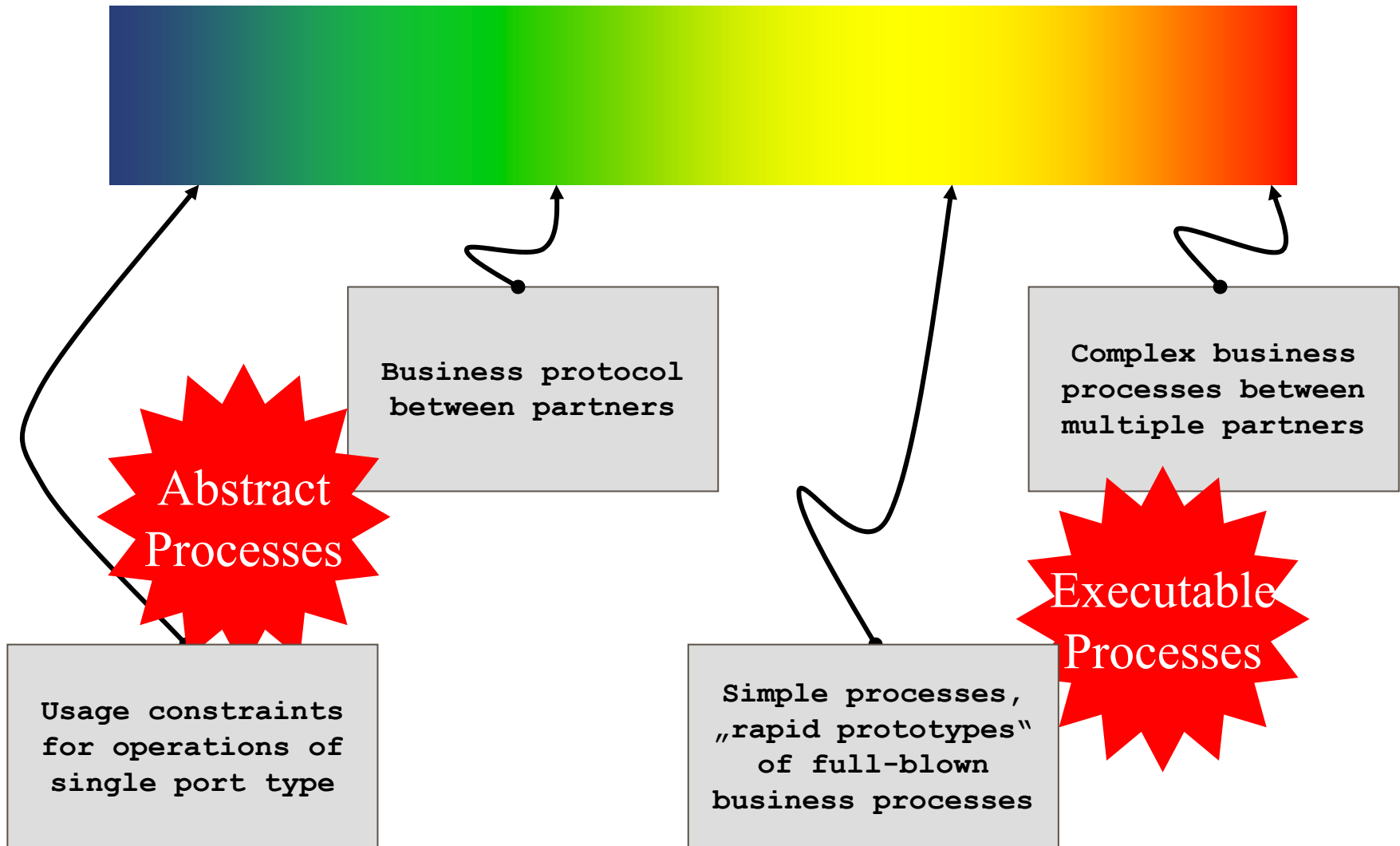


Flows And Web Services

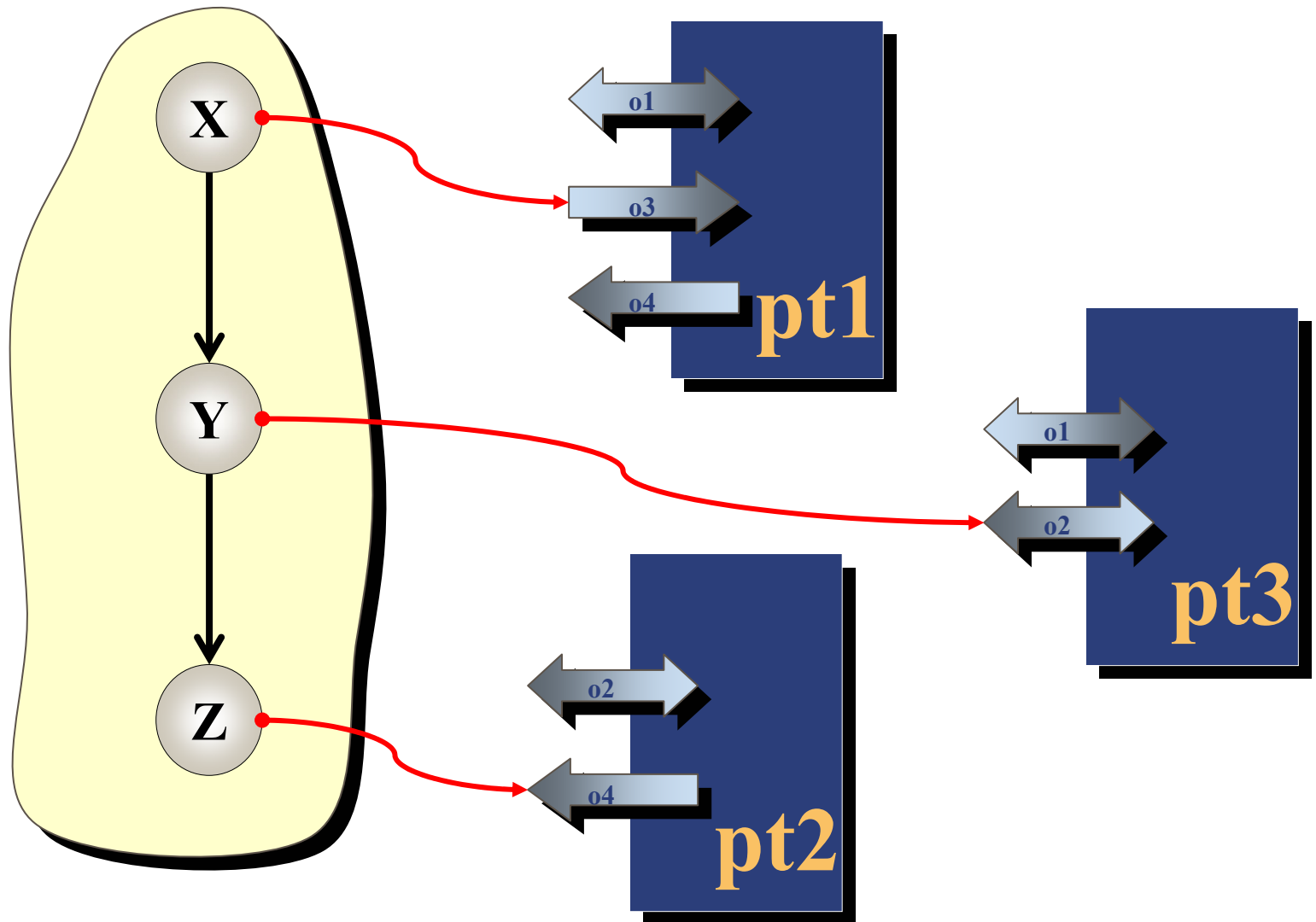
Business Processes = Flows



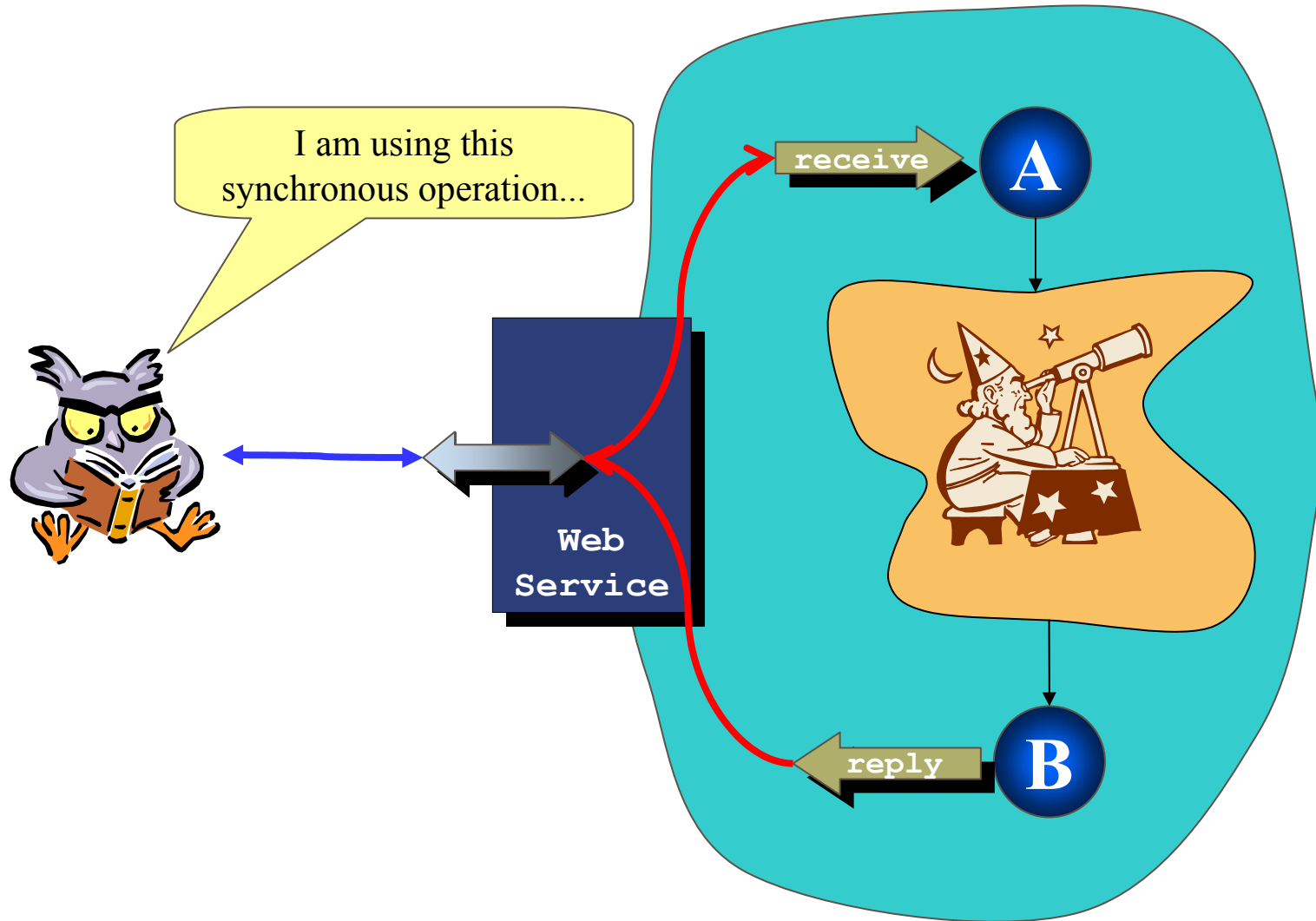
BPEL4WS: Usage Spectrum



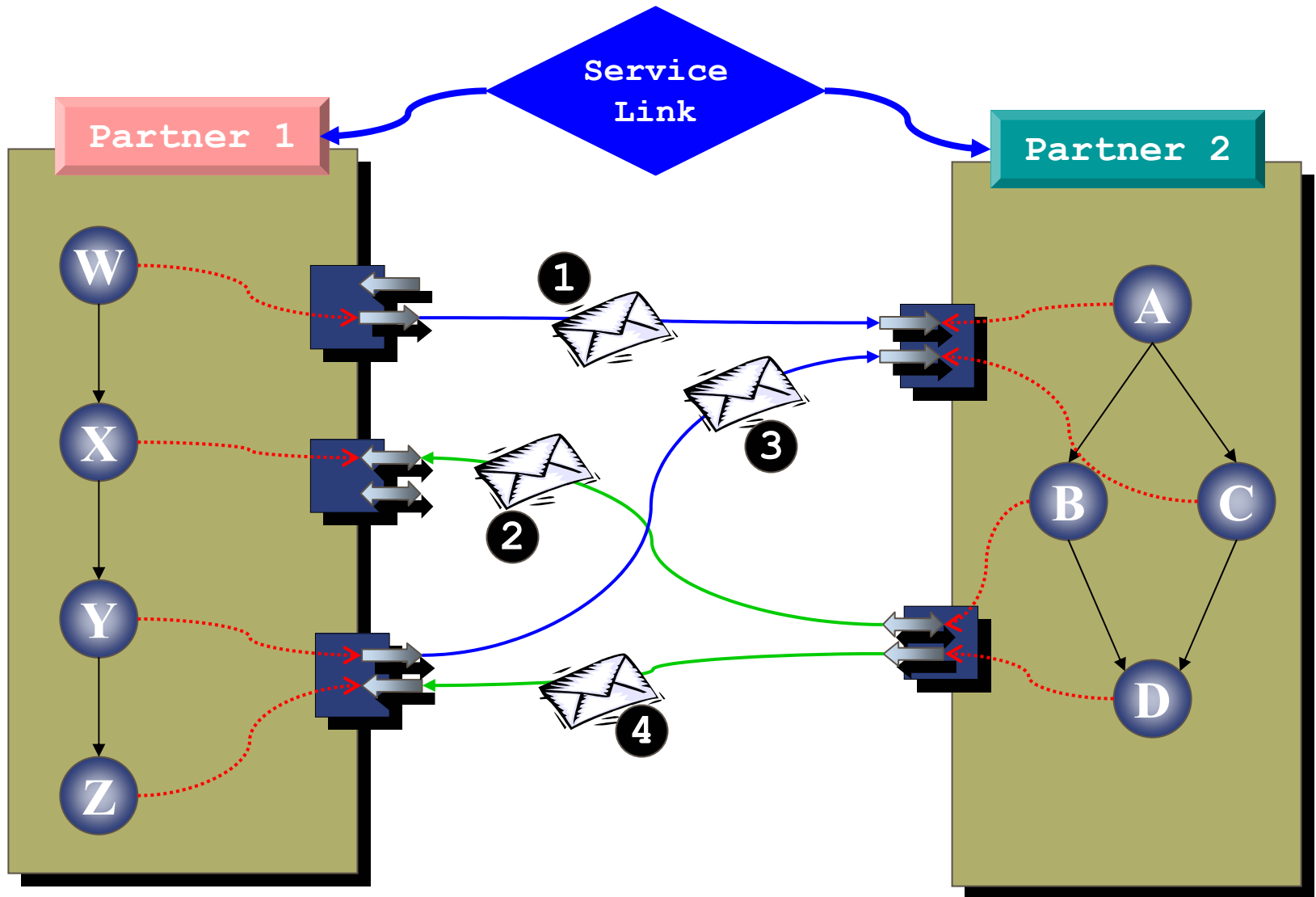
Business Processes Between Web Services



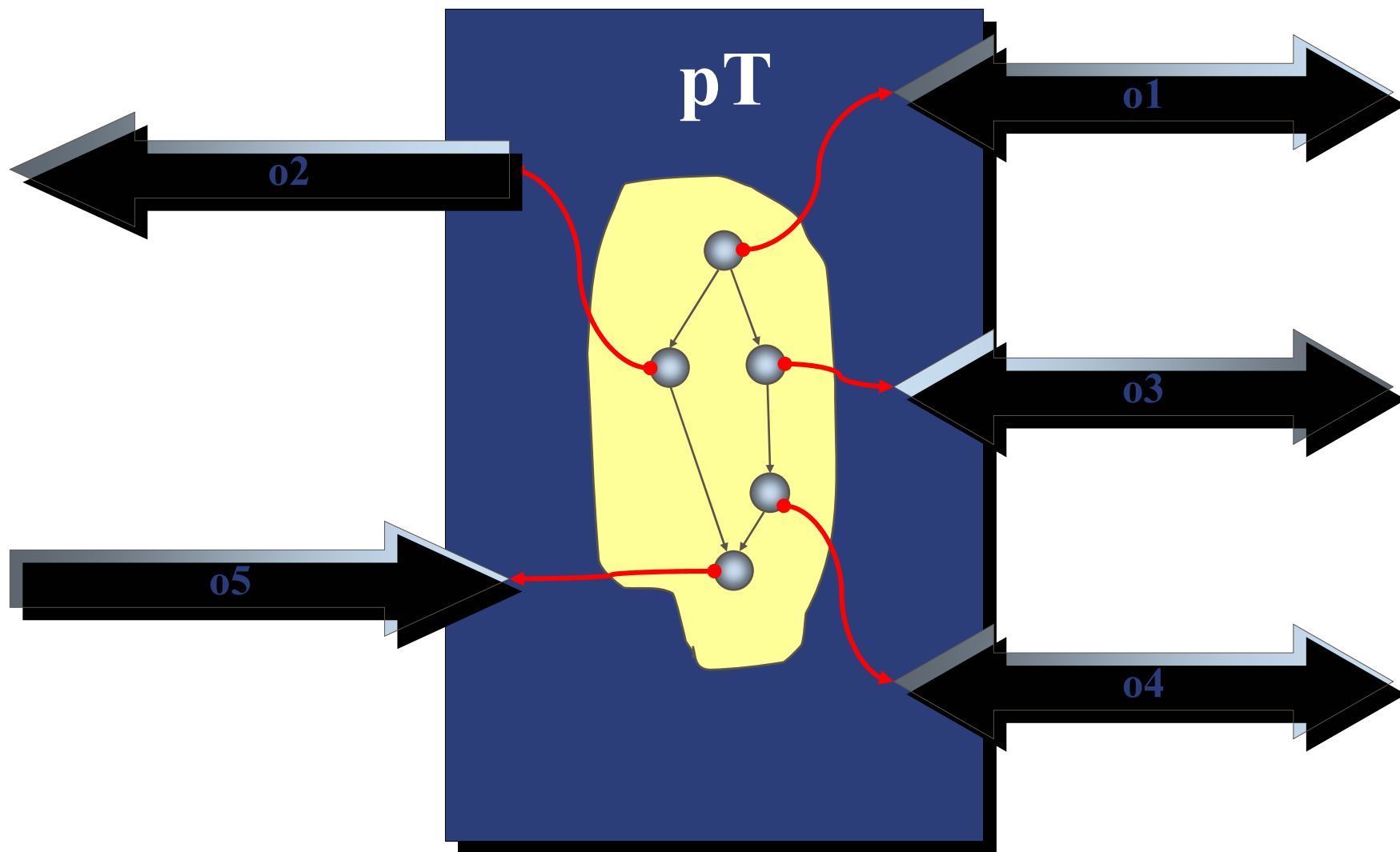
Business Processes As Web Services



Business Protocols

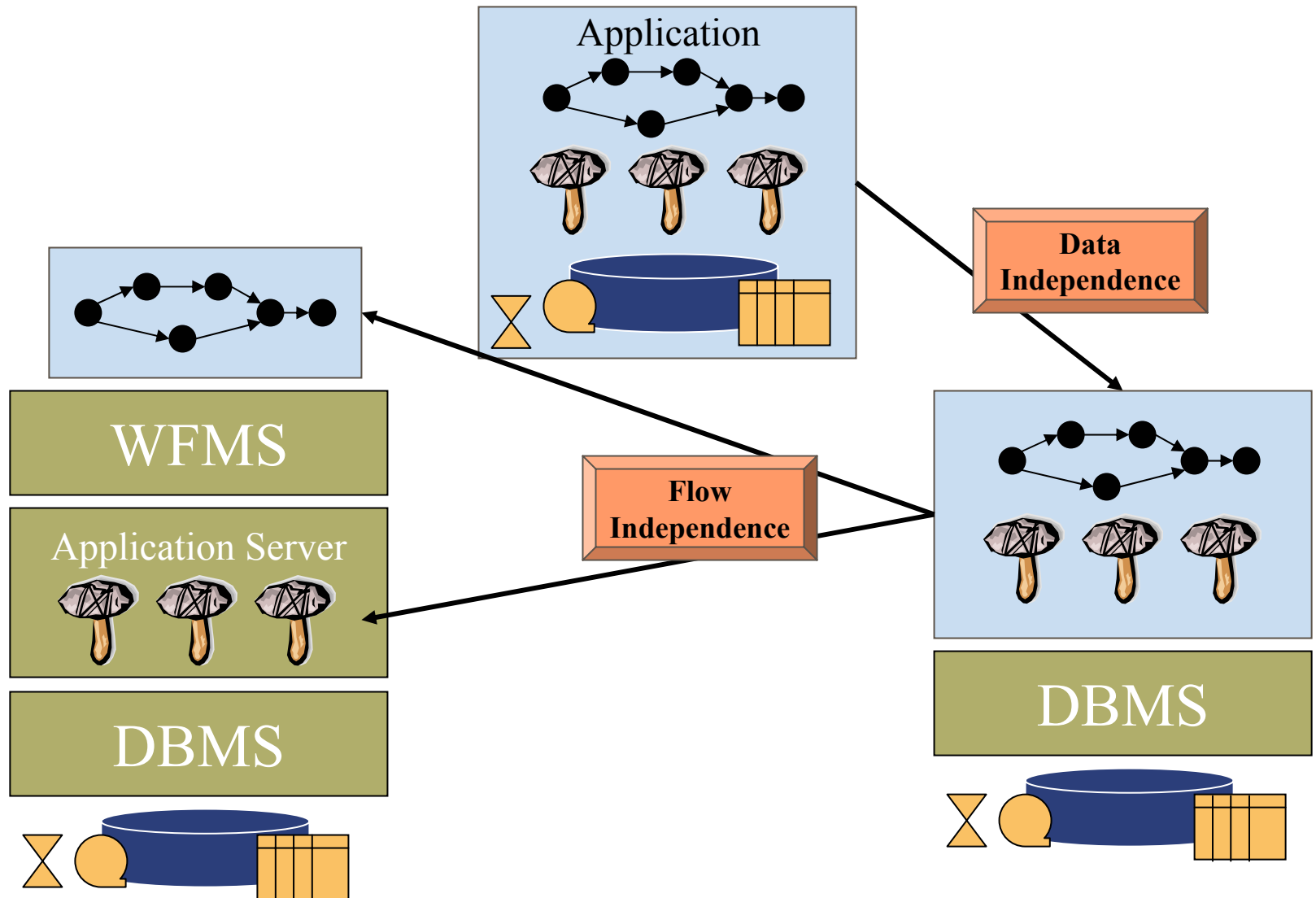


Constraining Invocation Orders

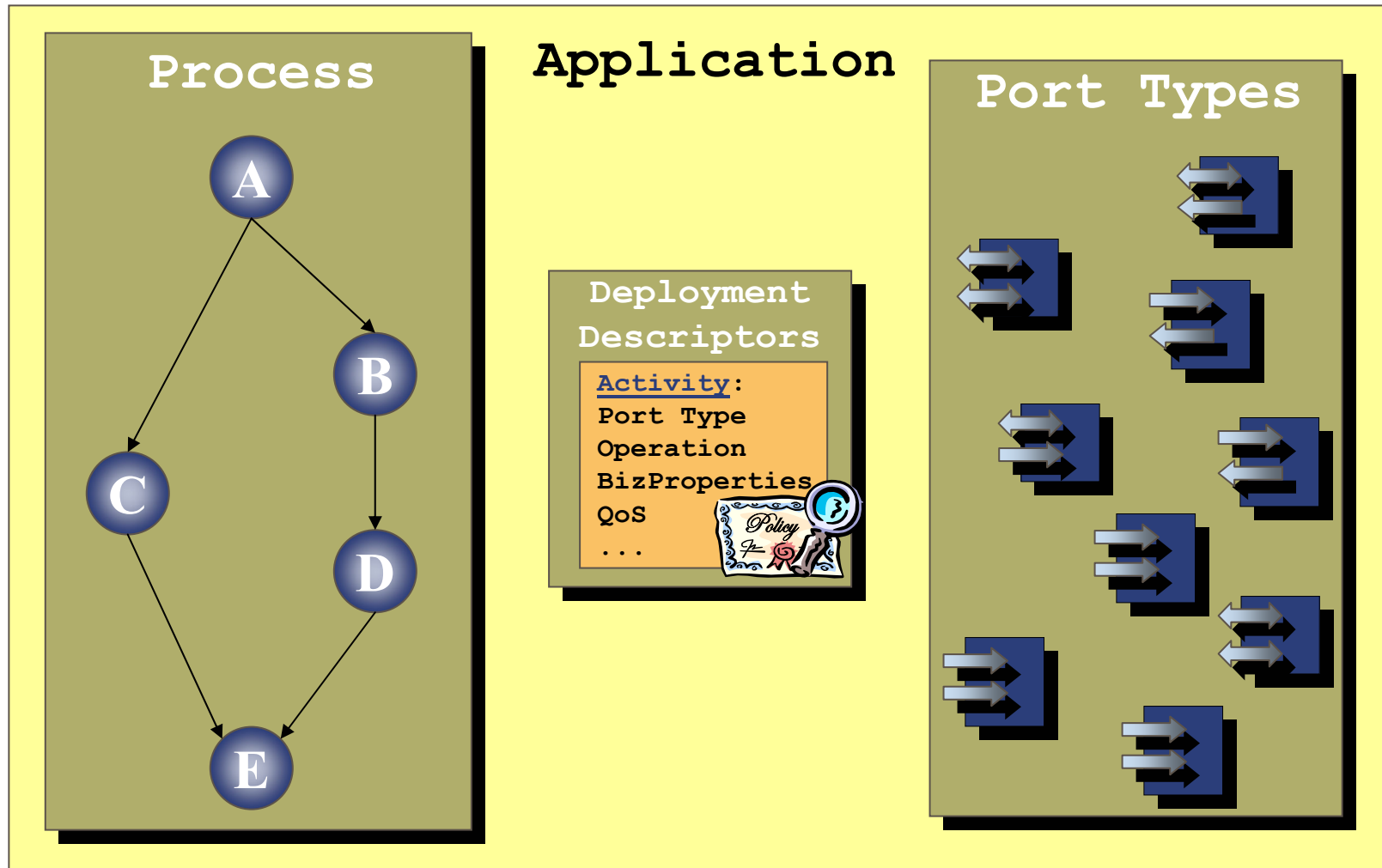


Two-Level Programming

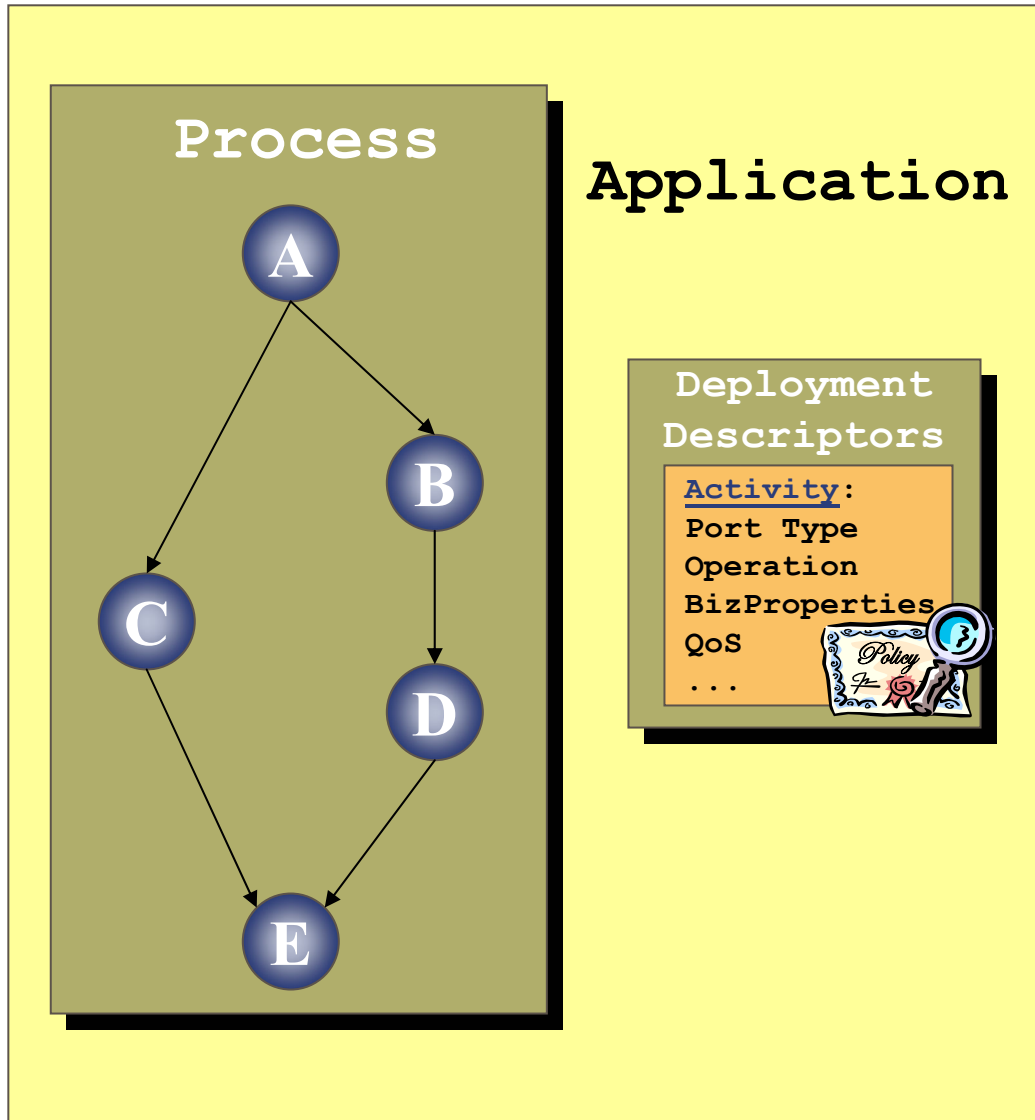
Why Using Flows For Application Construction?



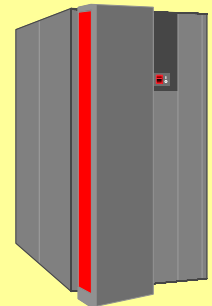
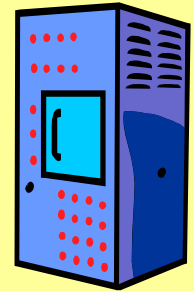
SOA Application Structure



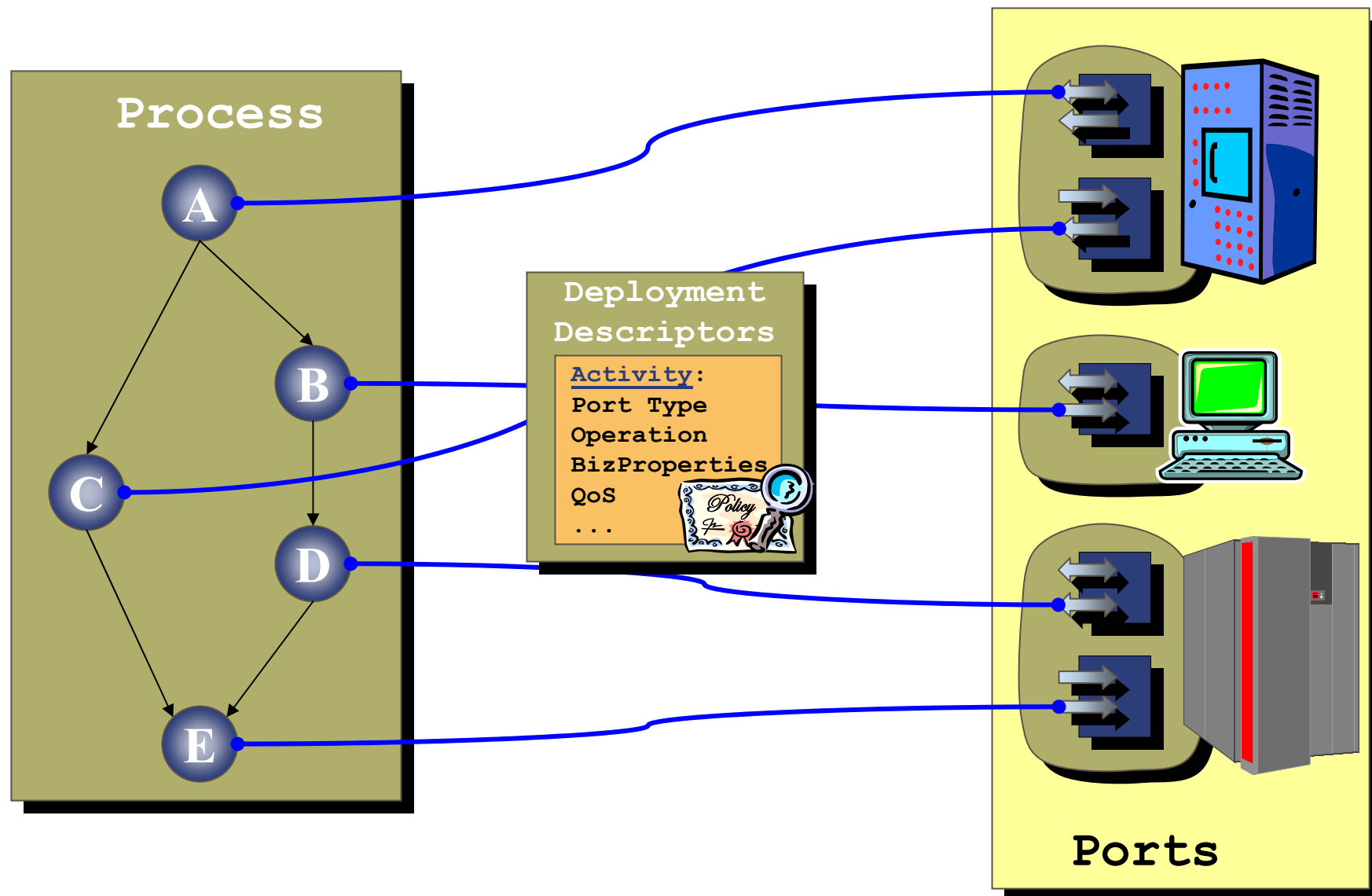
SOA Applications And The Grid



Grid (Business or OGSA)

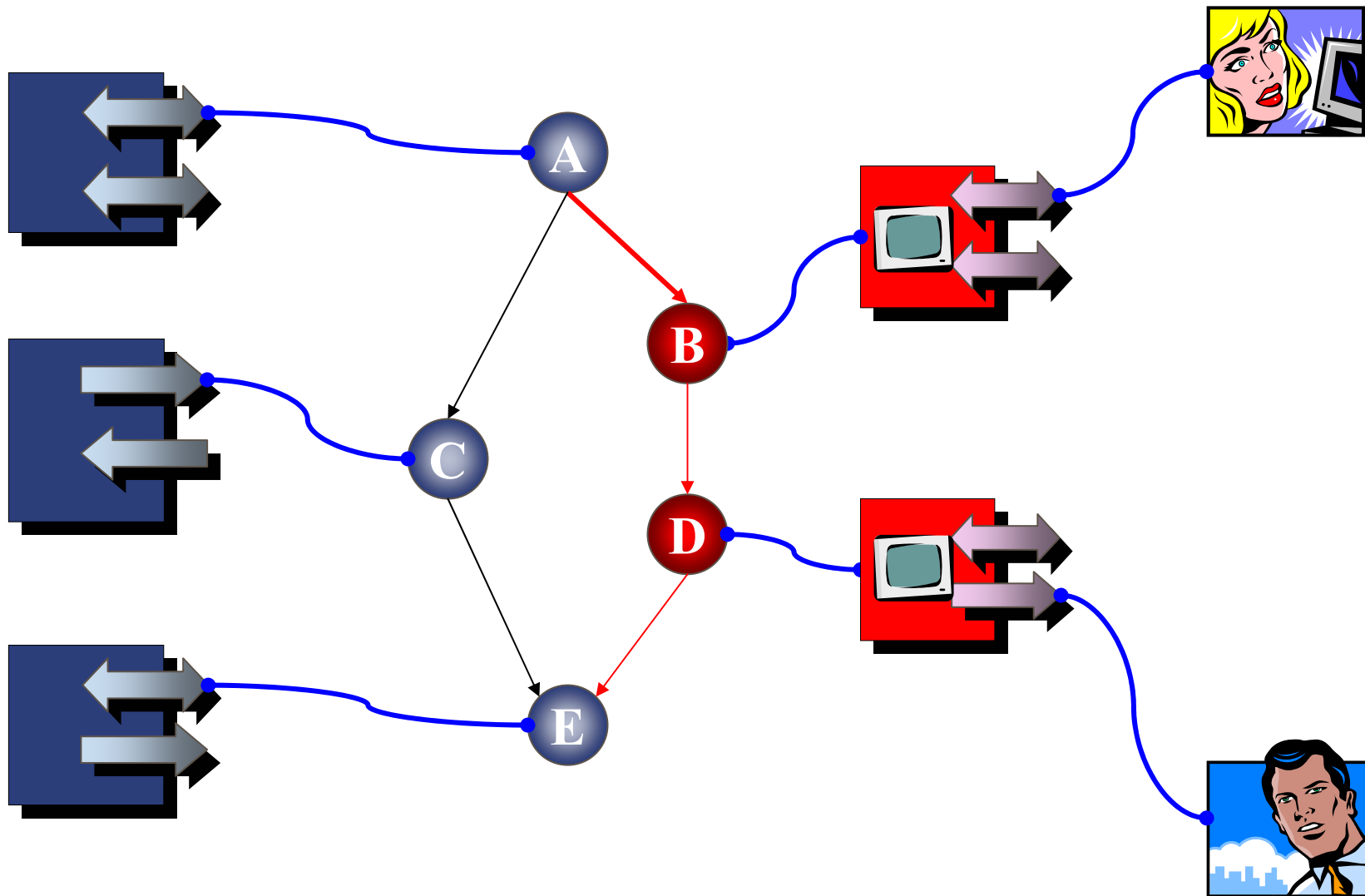


Deployment

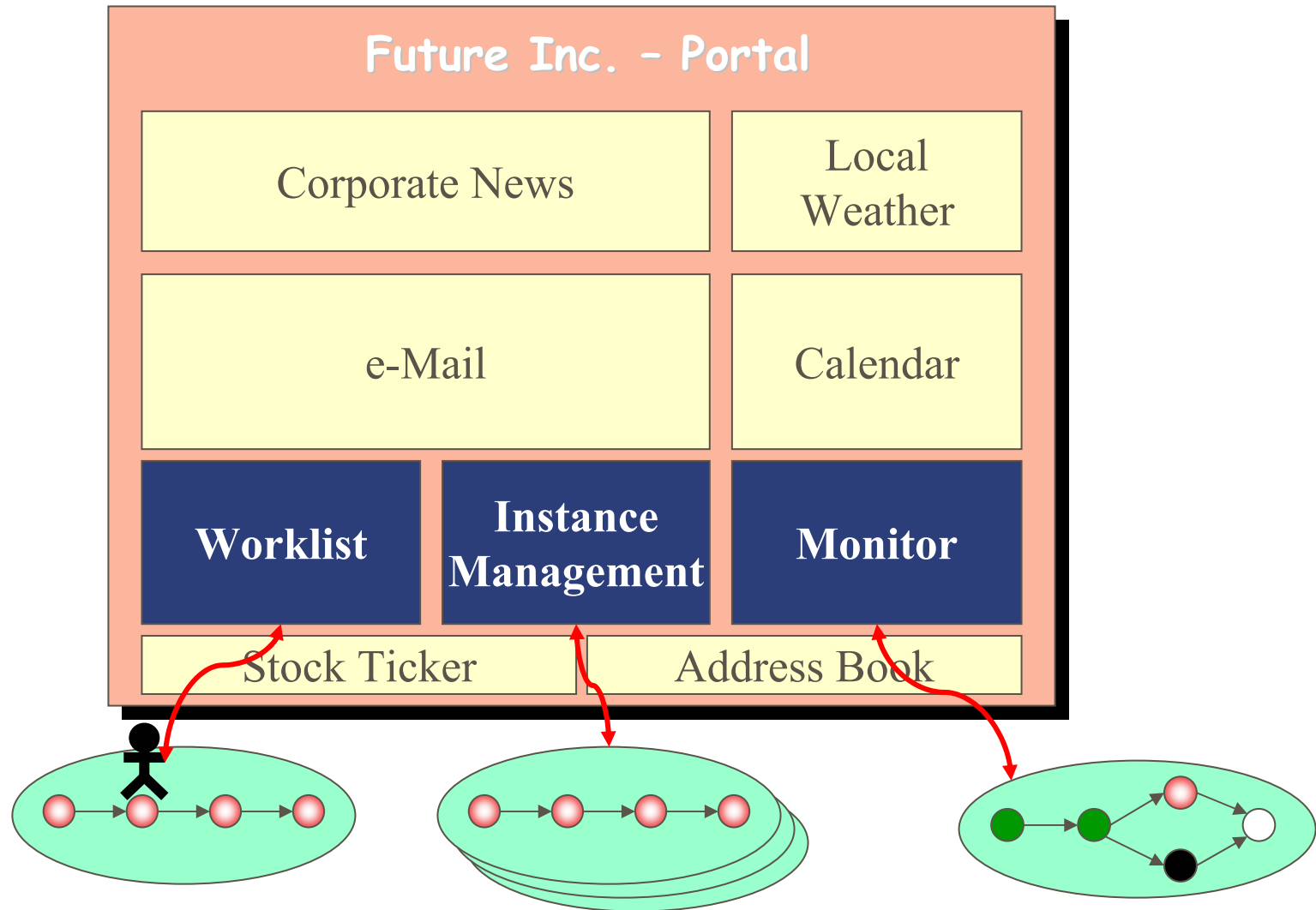


Involving People

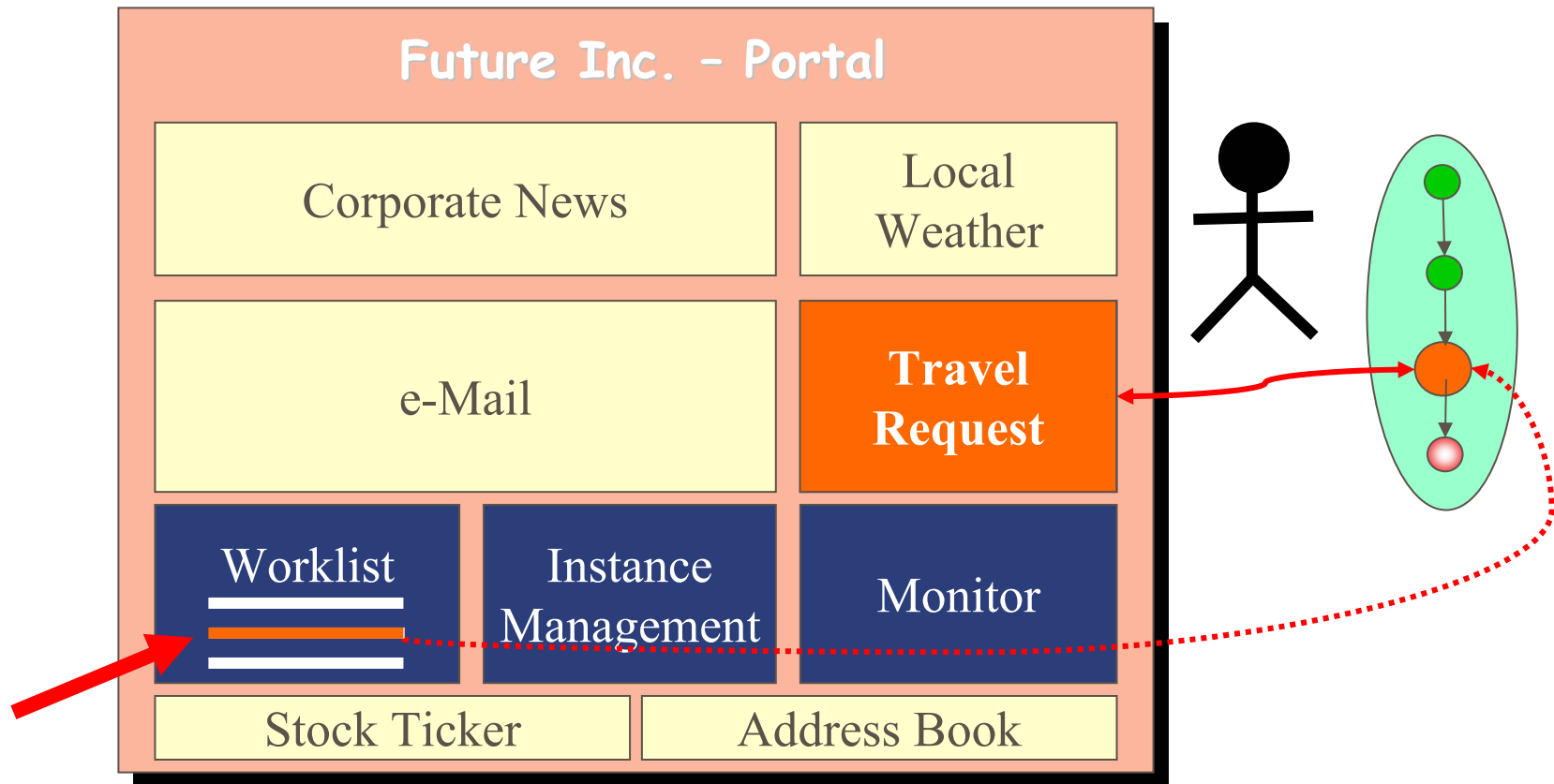
Exception Handling



Workflow Artifacts In Portals



Activities In Portals



Sample Work To-Be-Done...

To Dos

- Appropriateness of different techniques for user-facing interactions
 - Flow-based dialogs, Struts, JSF,...
- Business process definitions and Semantic Web
- Flows and Grid
 - Flow as scheduling language, “JCL”
- Process templates
 - How to customize processes or extend process fragments without changing semantics of “main” process?

Agenda

Introduction

Virtual Components

Virtual Environments

Application Structure

Aggregations

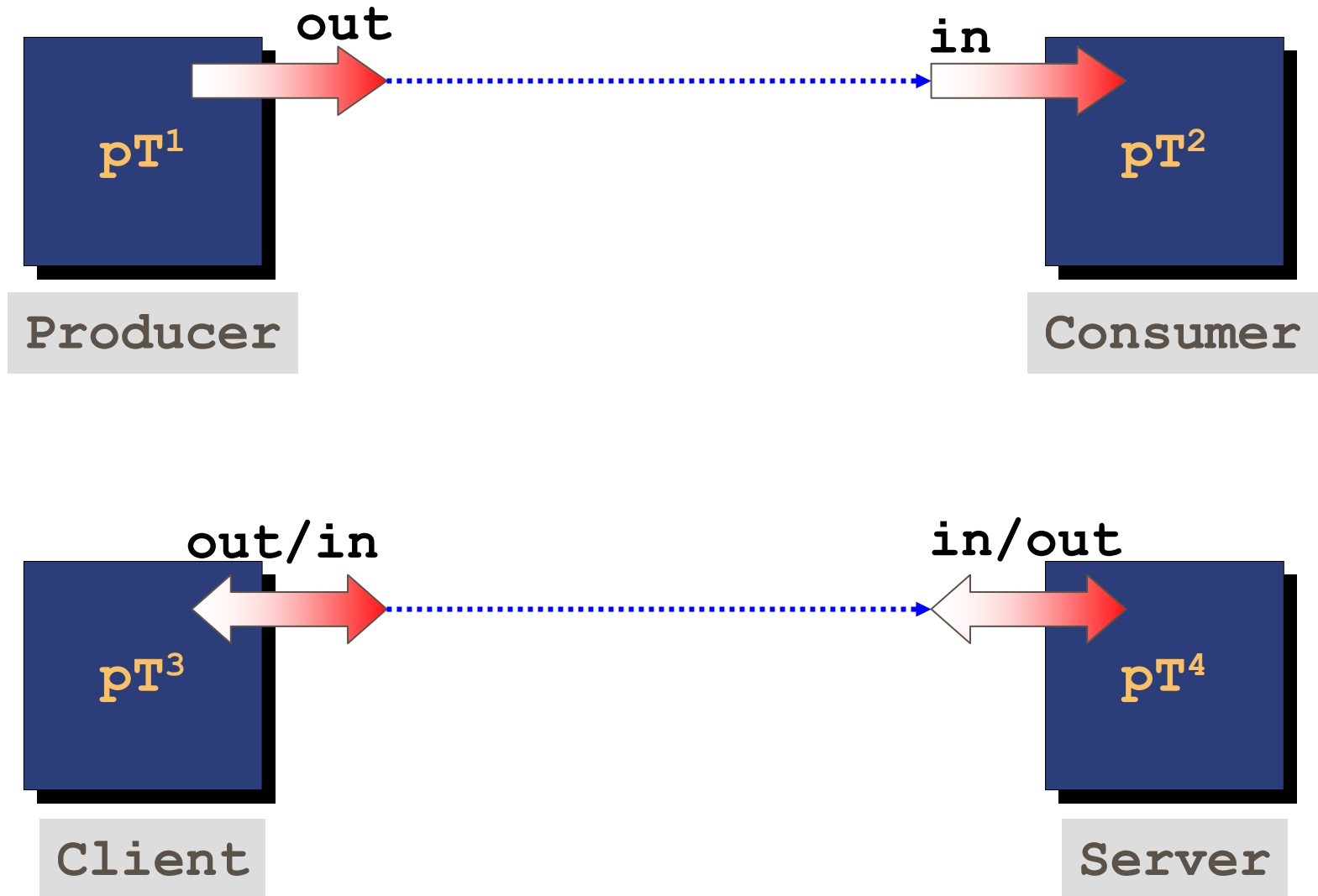
Summary & Conclusion

Classifying Service Aggregations

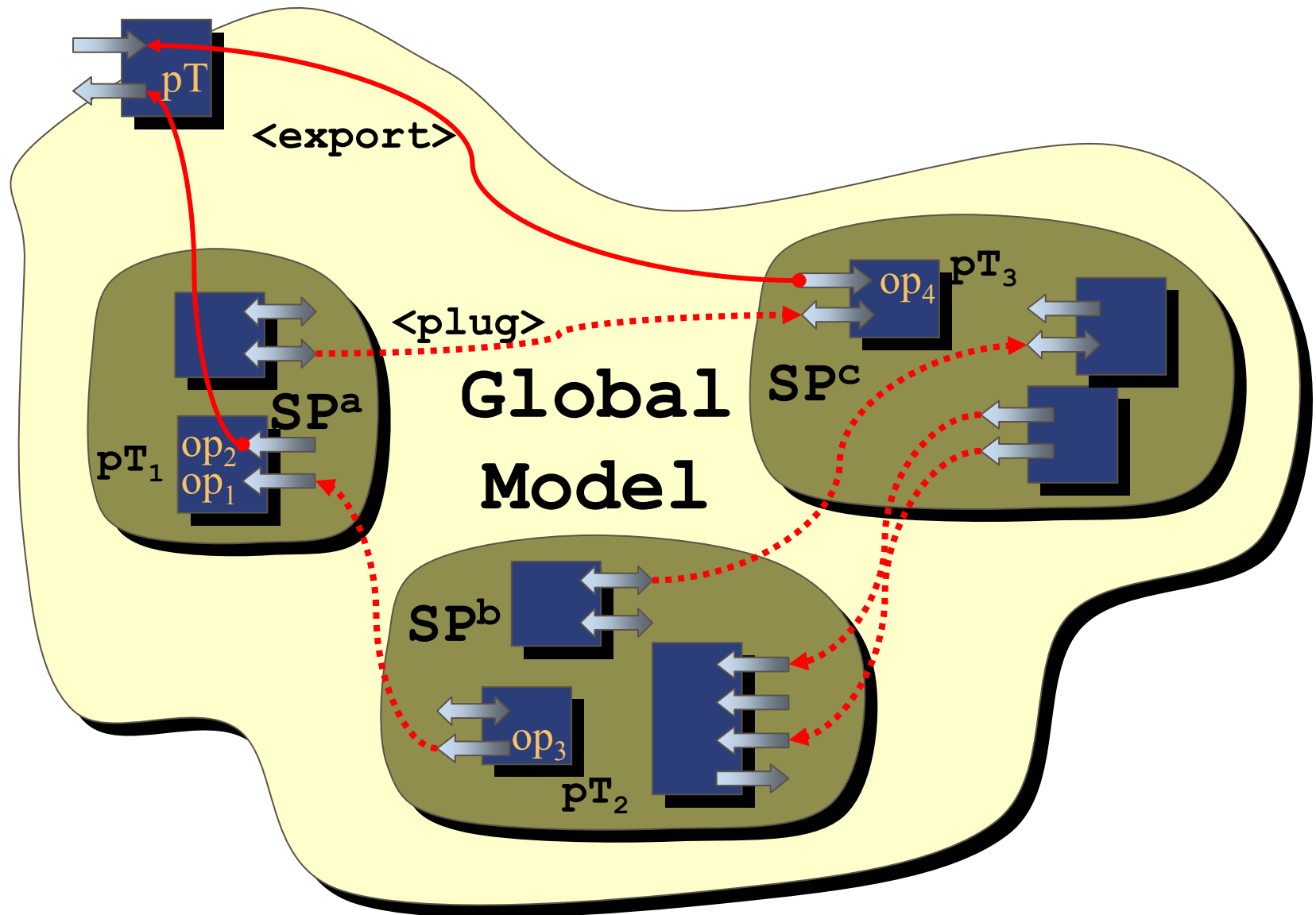
What? How?	Types	Instances
Constrained	Choreography MEP	Agreement Service Domain
Unconstrained	Inheritance Service Types	

Global Models: Recursive Composition

Matching Transmission Primitives

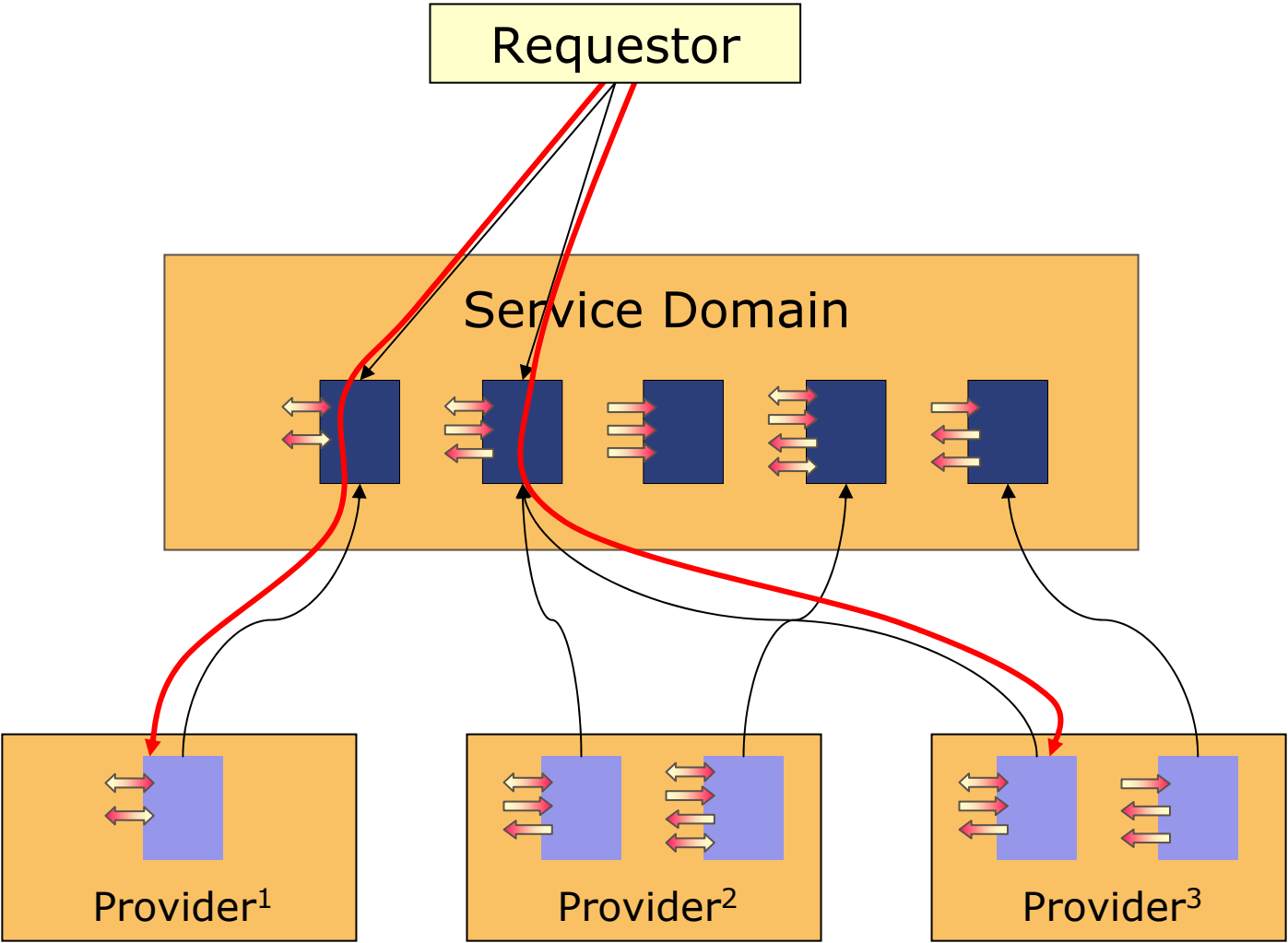


Global Models: Recursive Composition

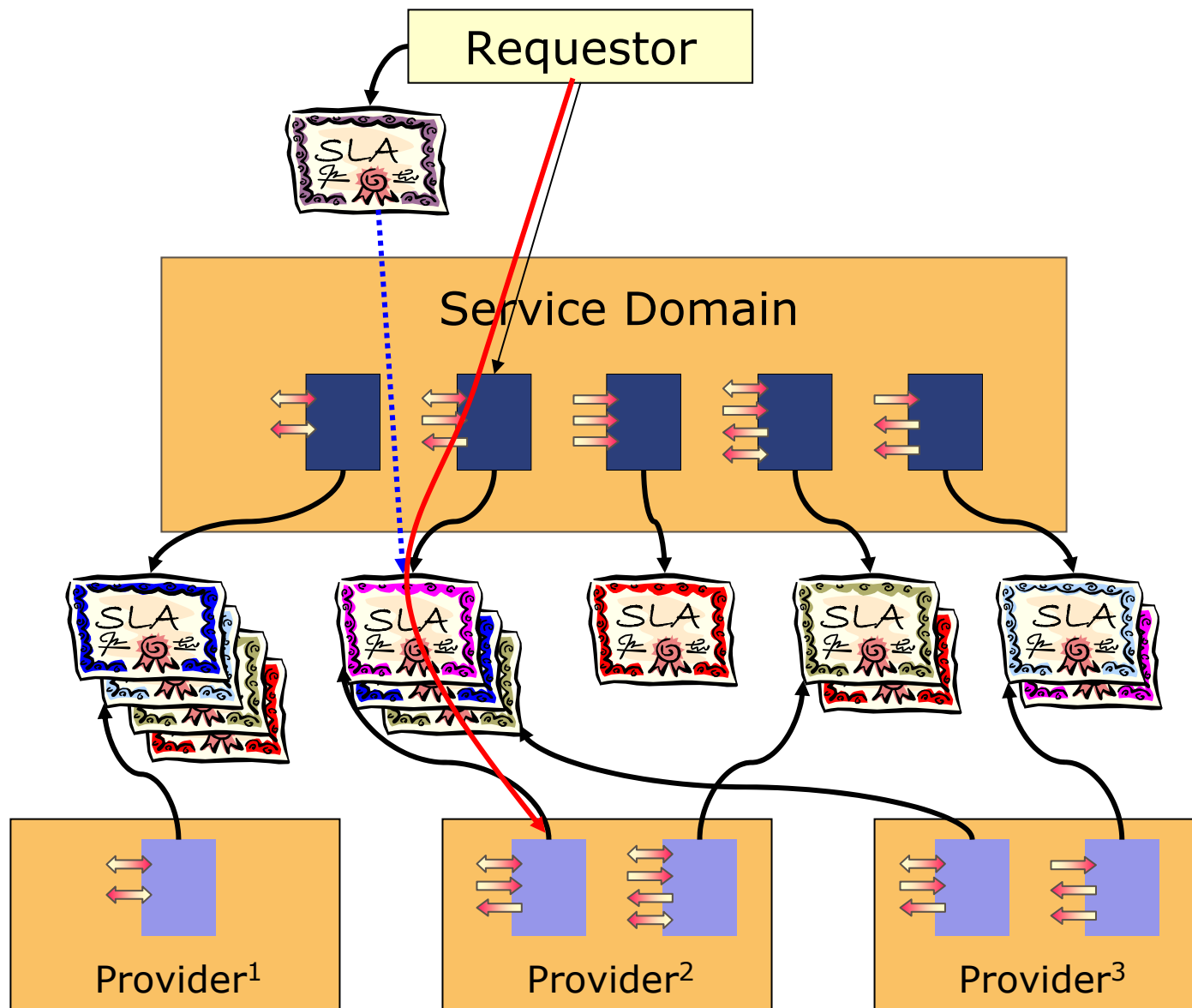


Service Domains

Service Domain

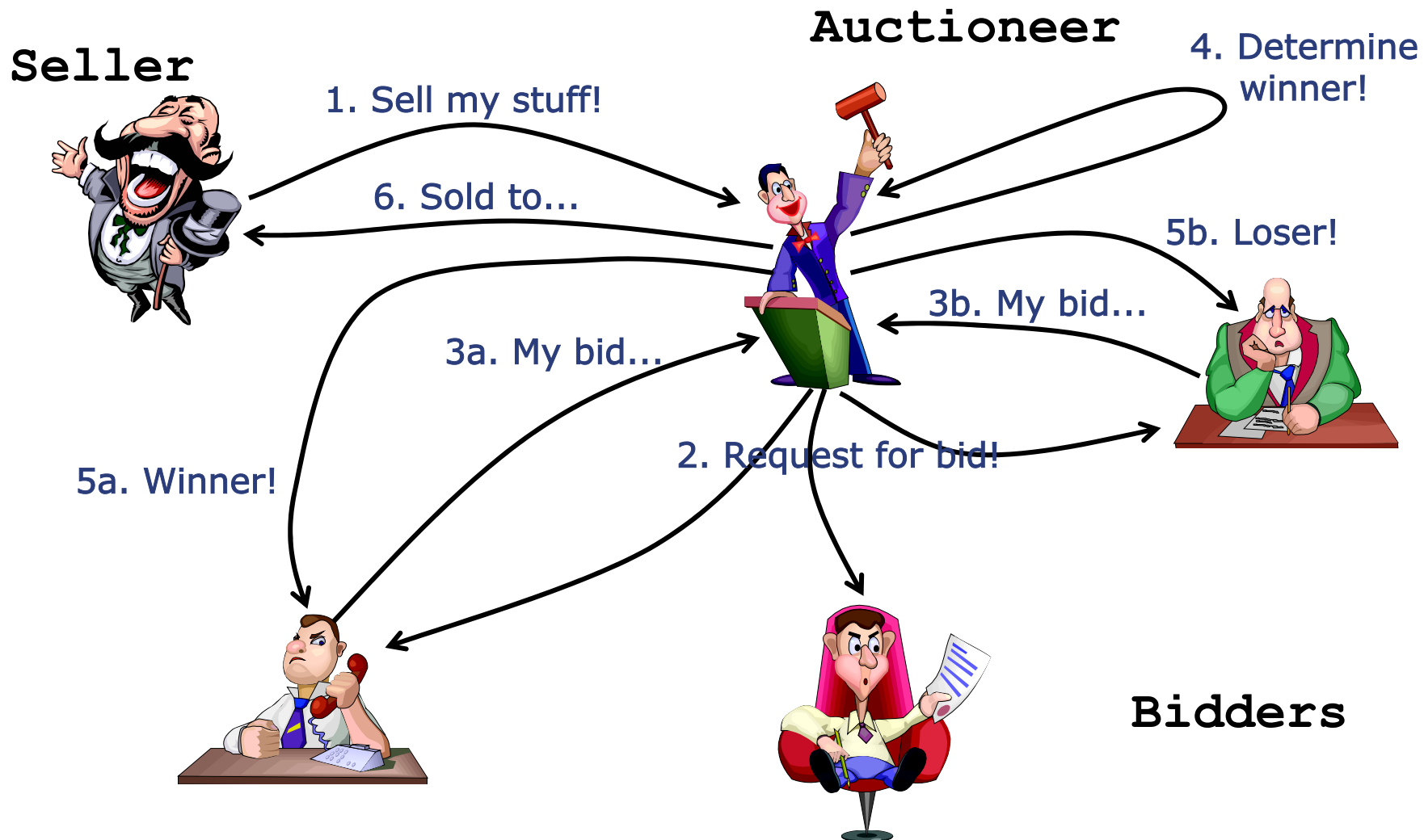


Service Domain And SLAs

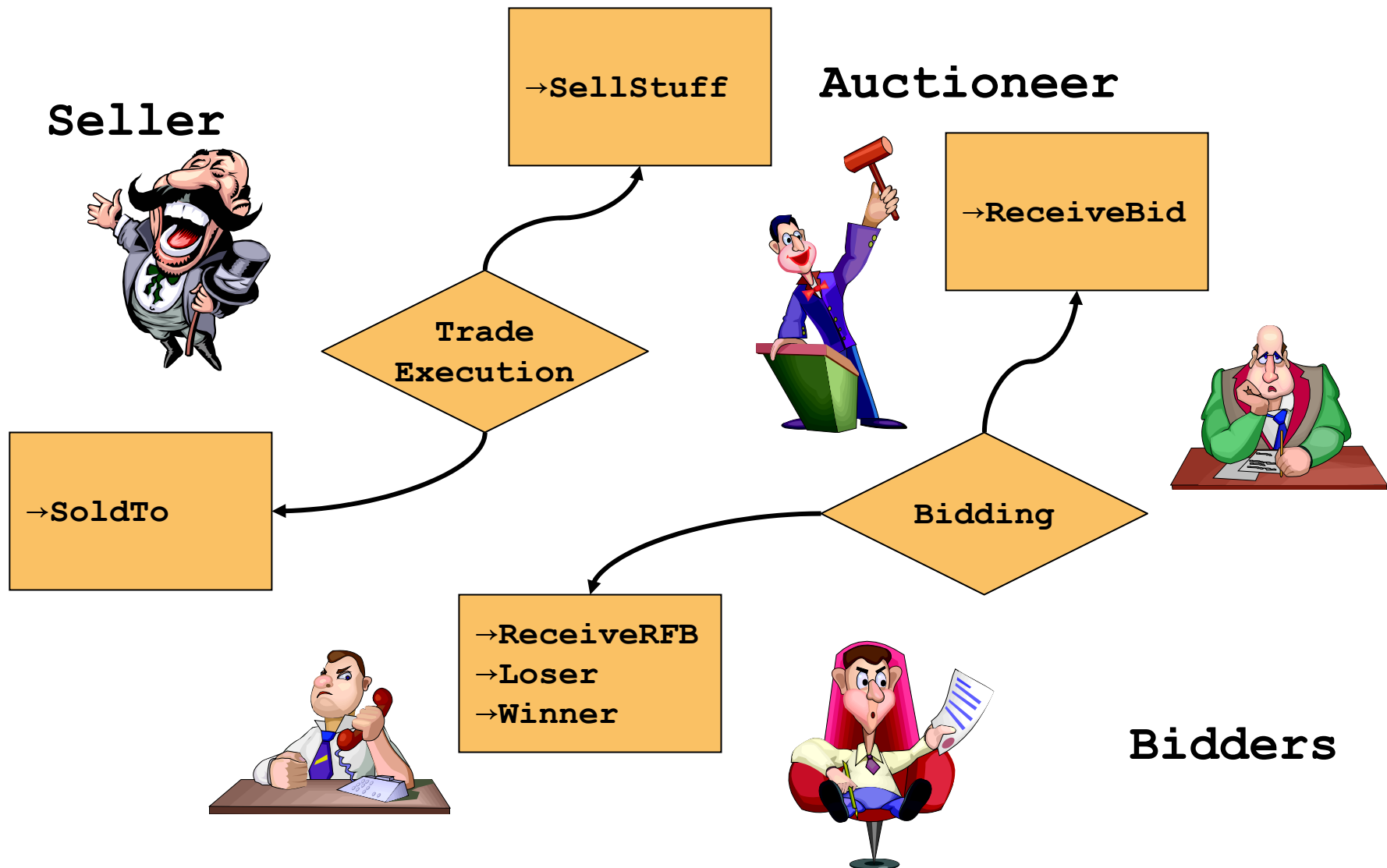


Coordination: Out of The Galaxy

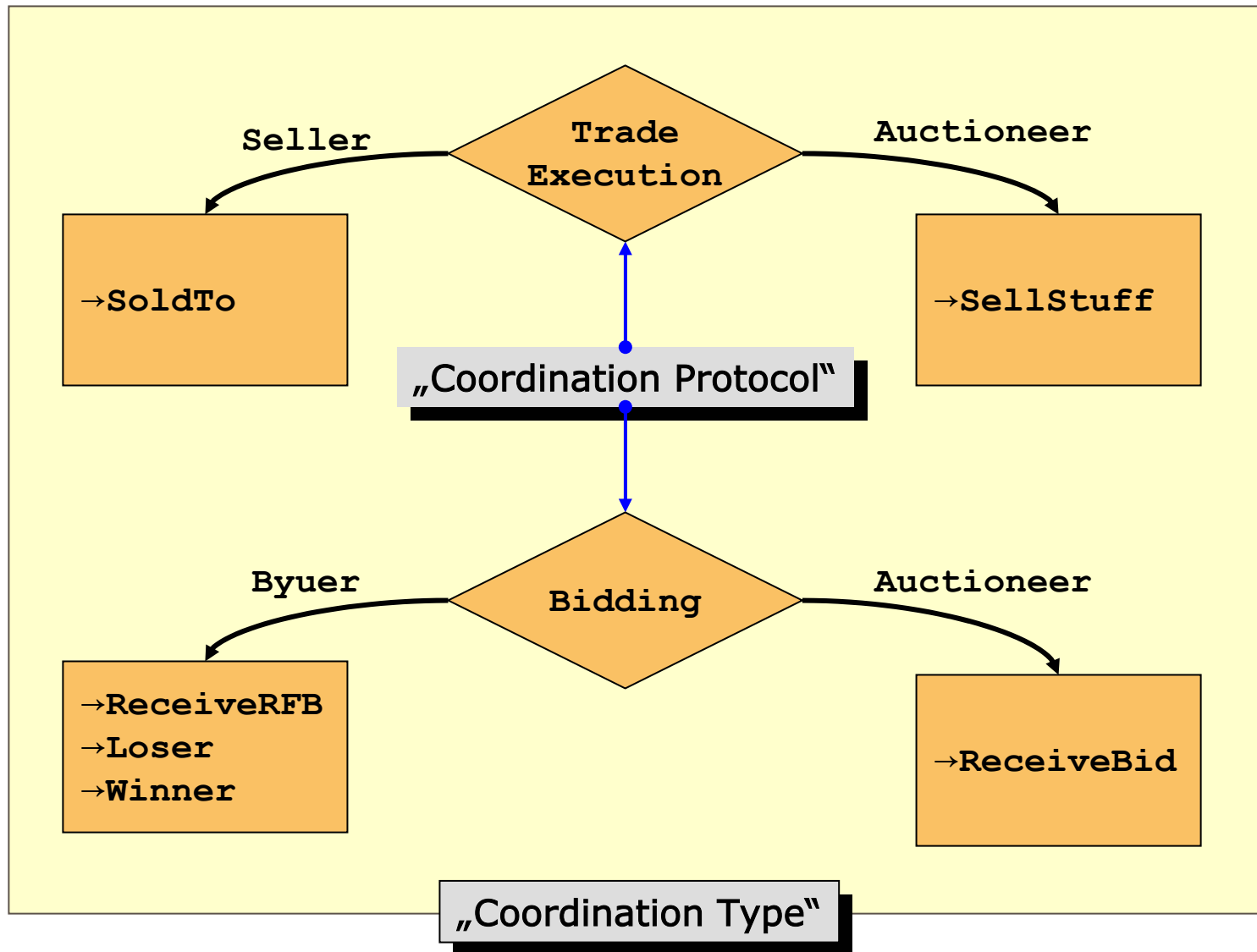
An Example



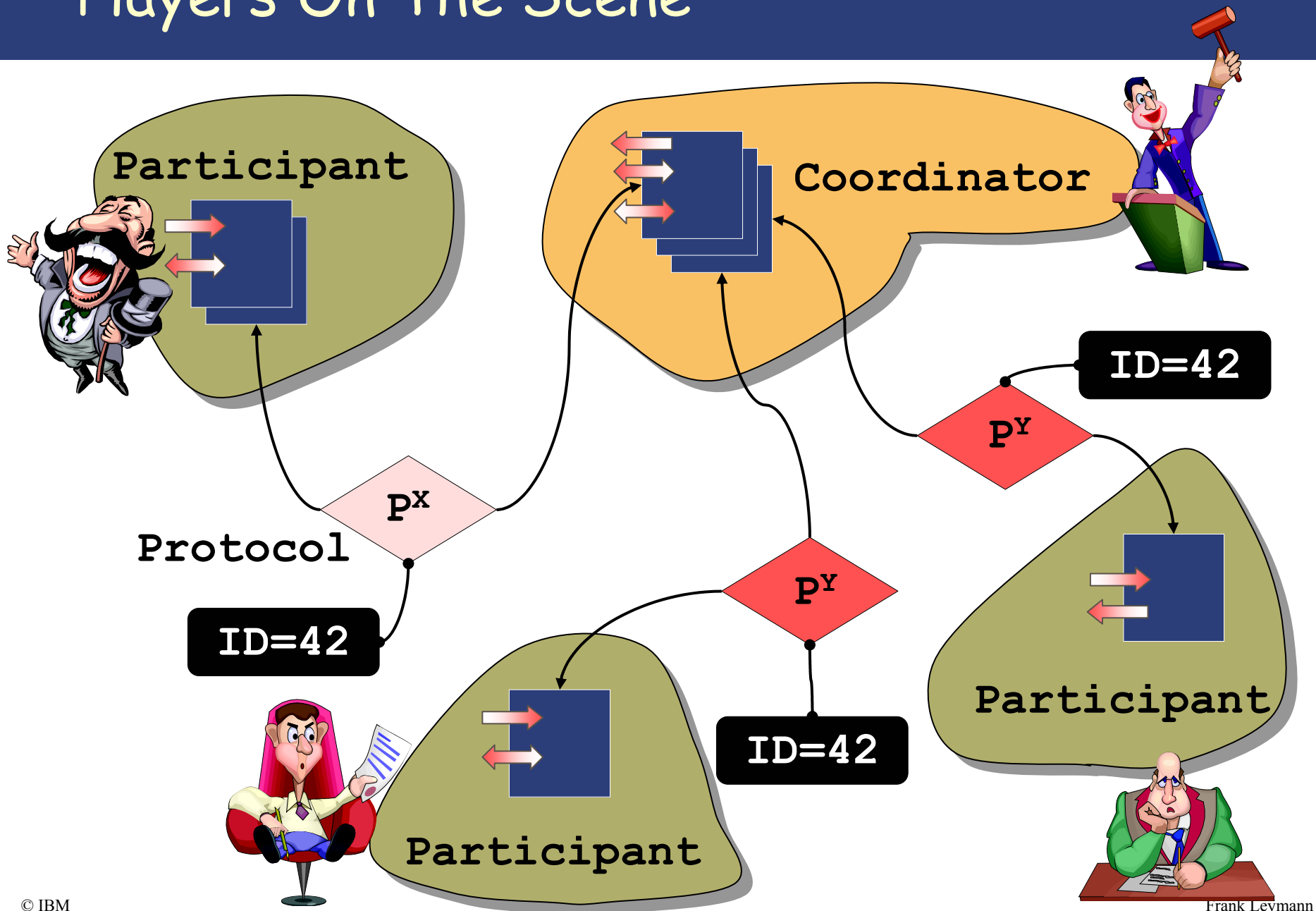
Messages Exchanged



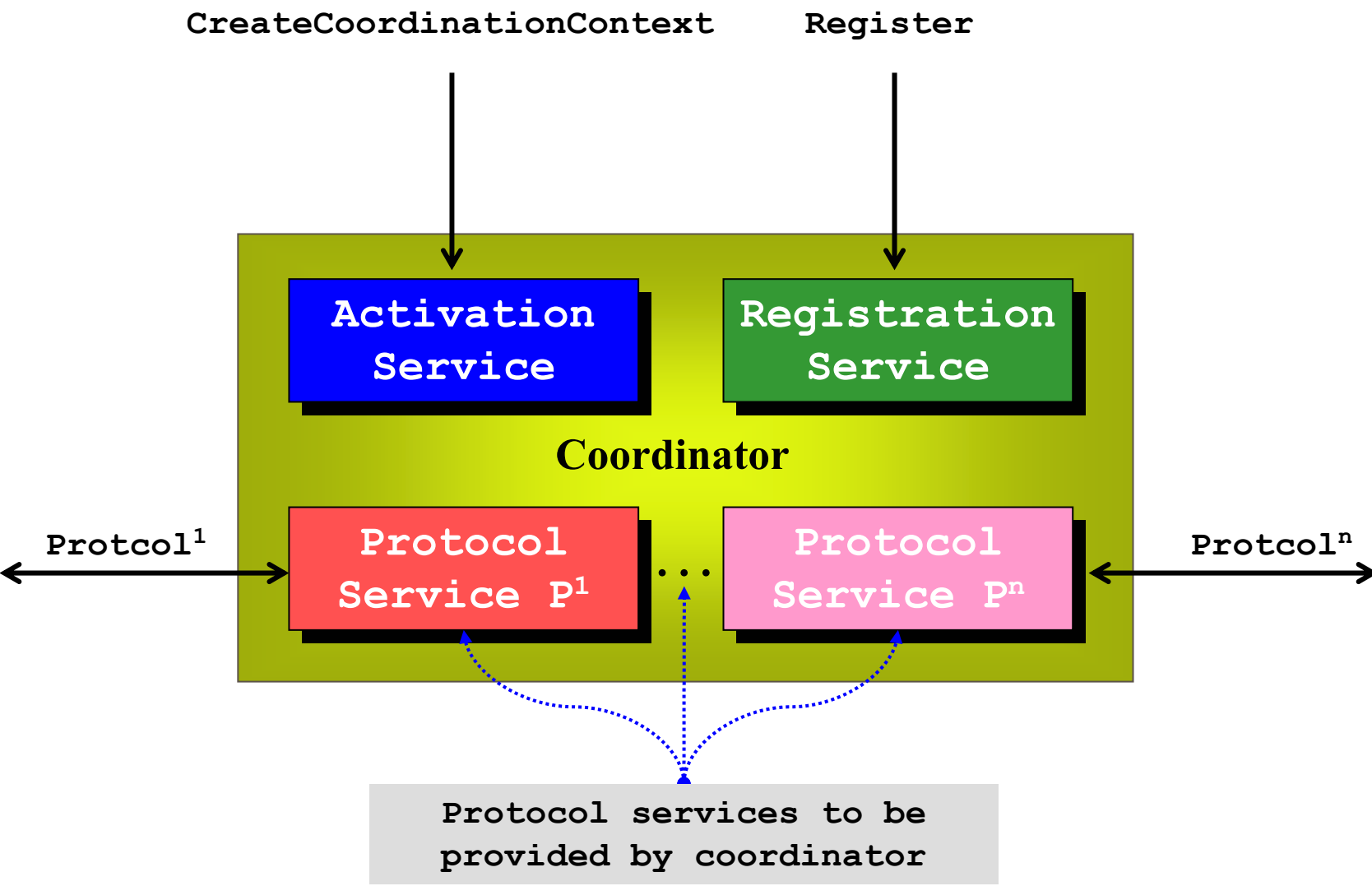
Coordination Protocols And Types



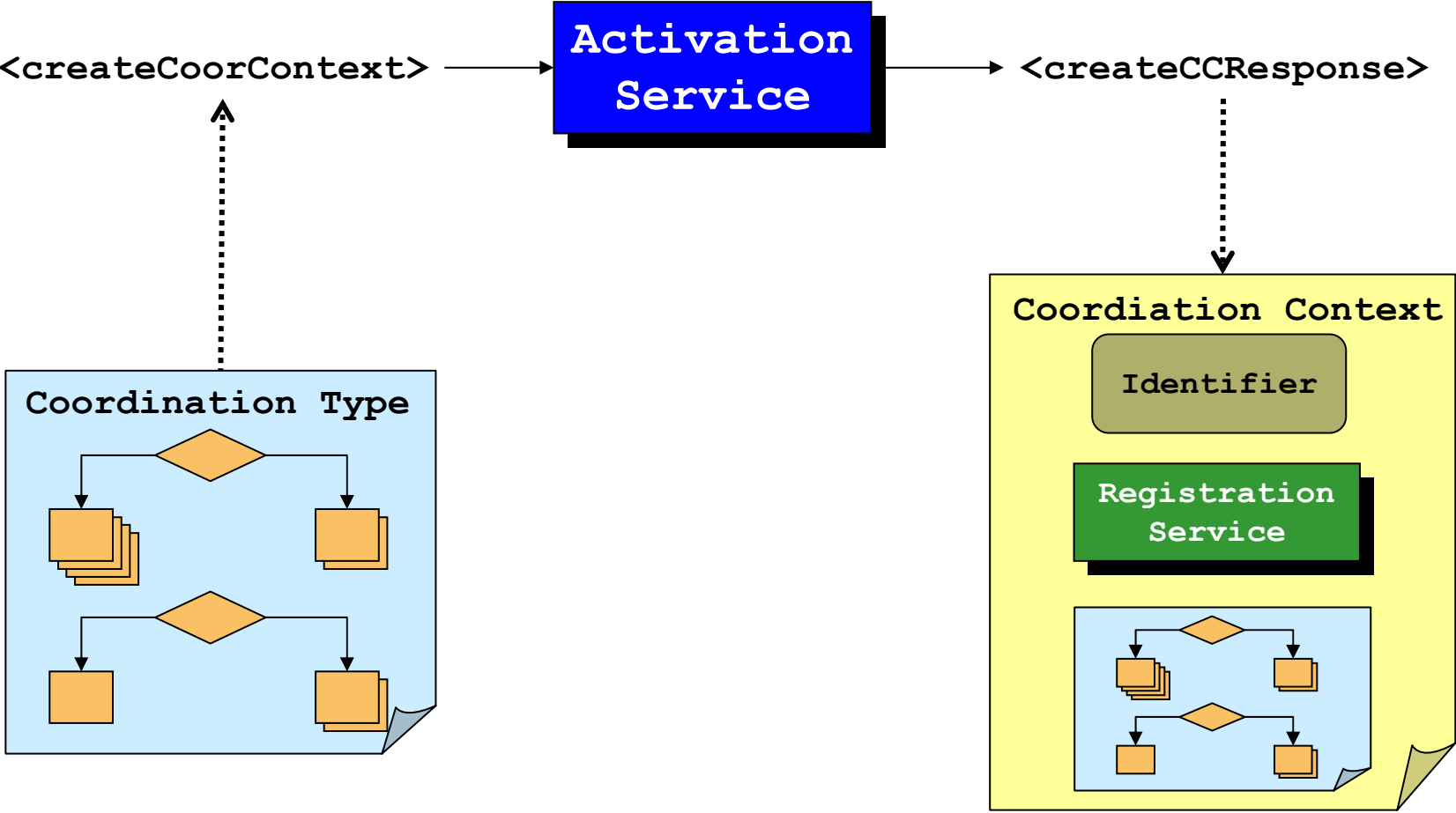
Players On The Scene



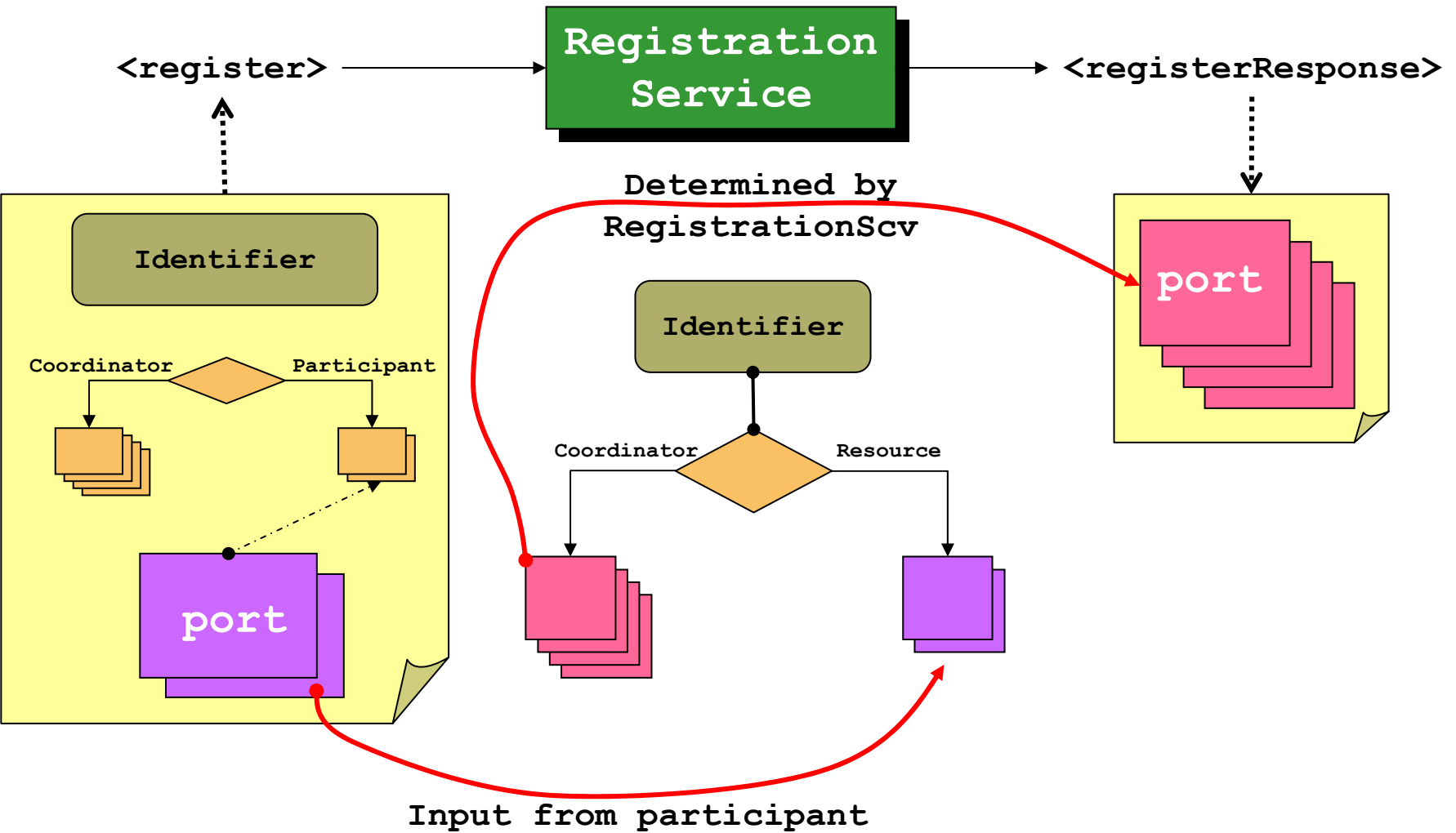
...And How They Relate



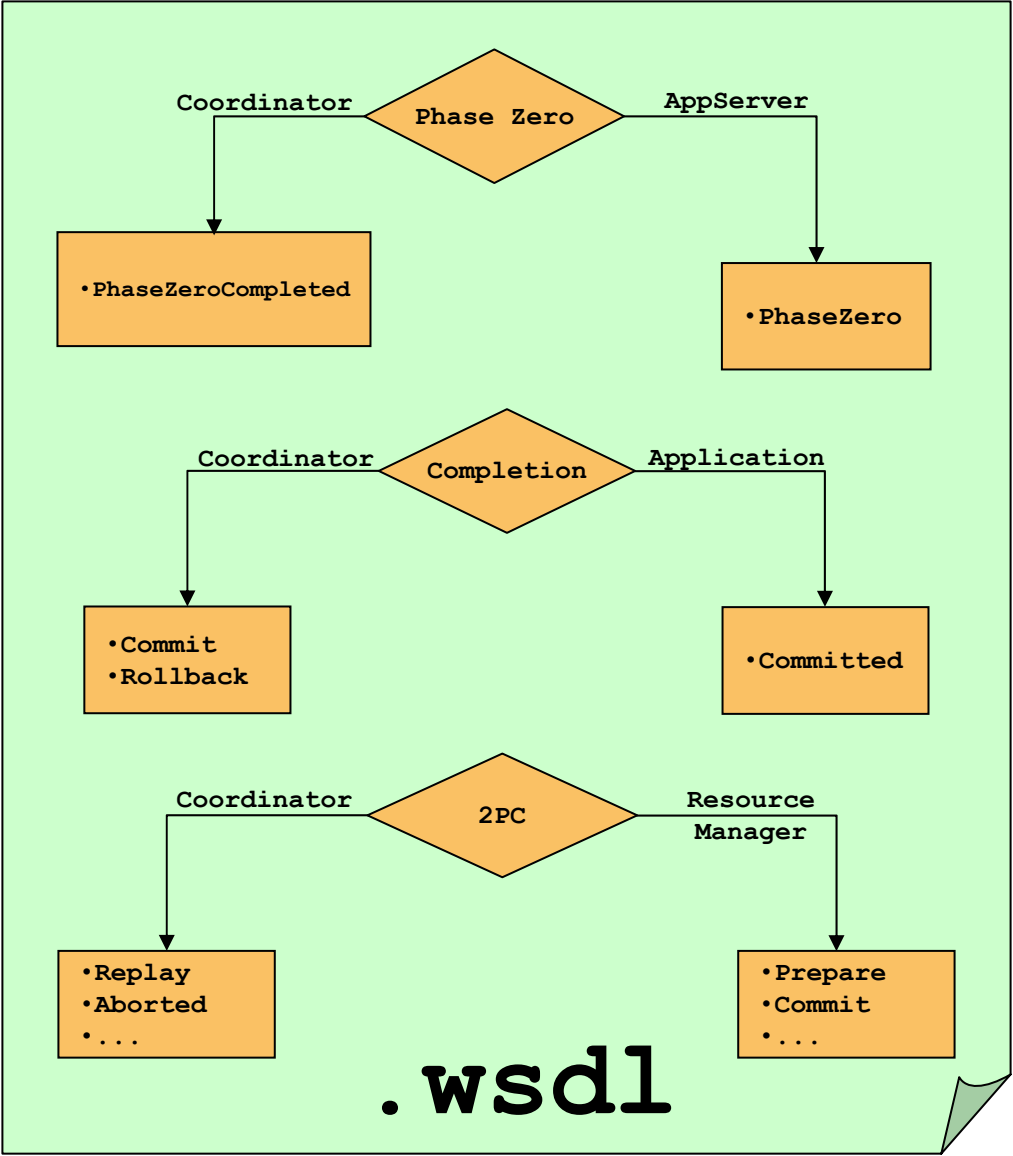
Creating An Activity



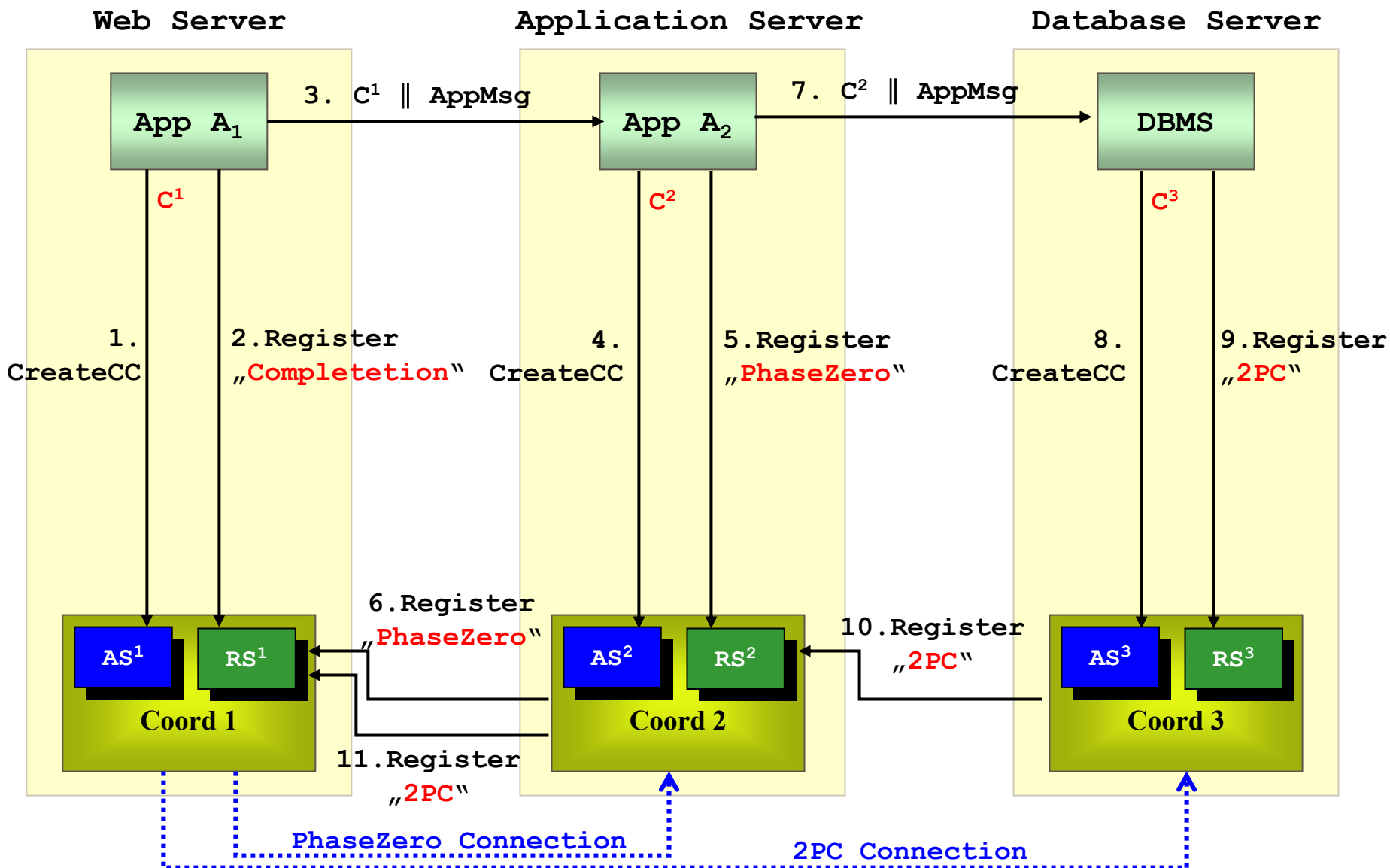
Registering With An Activity



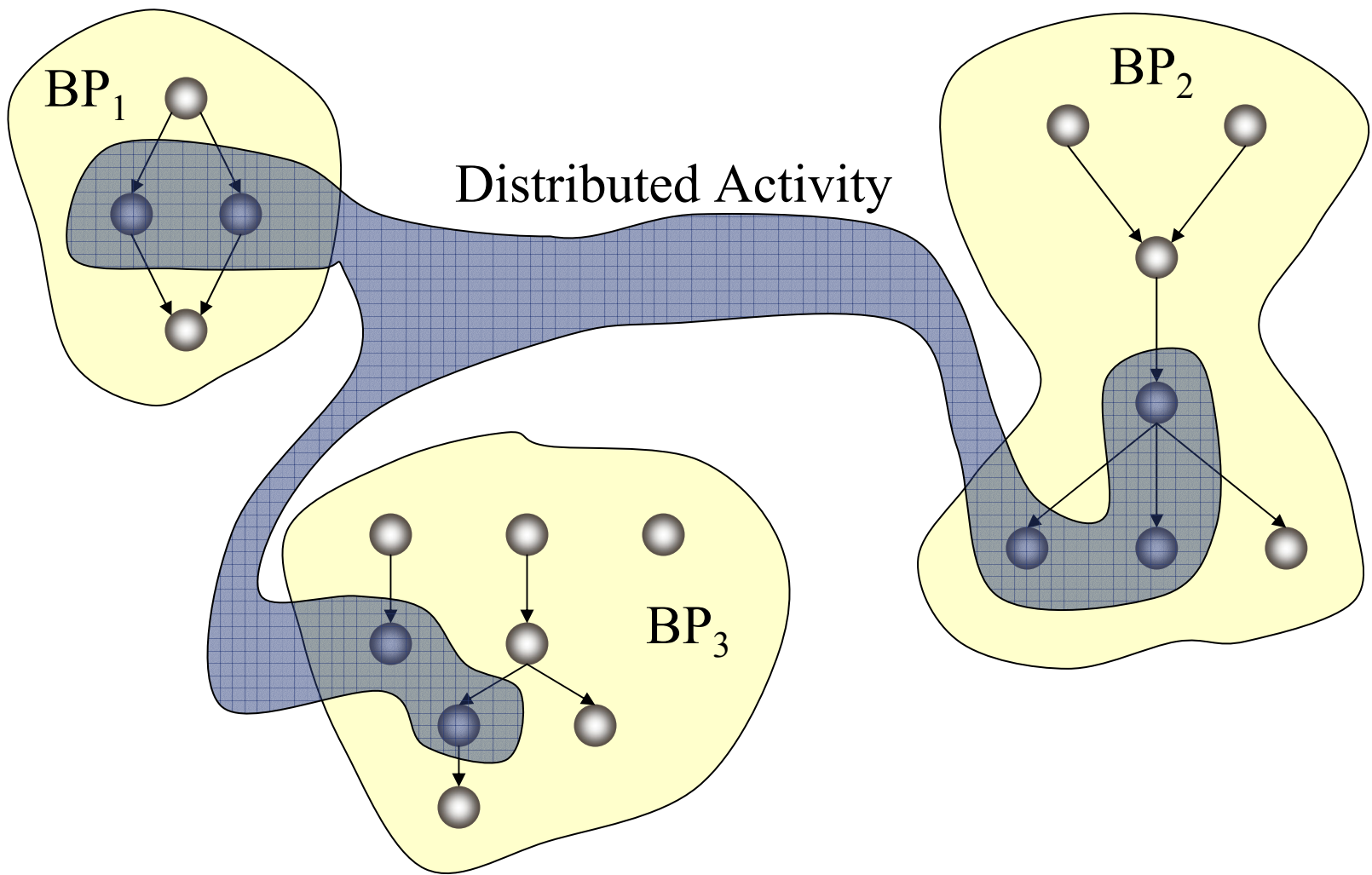
Example Coordination Type: 2PC



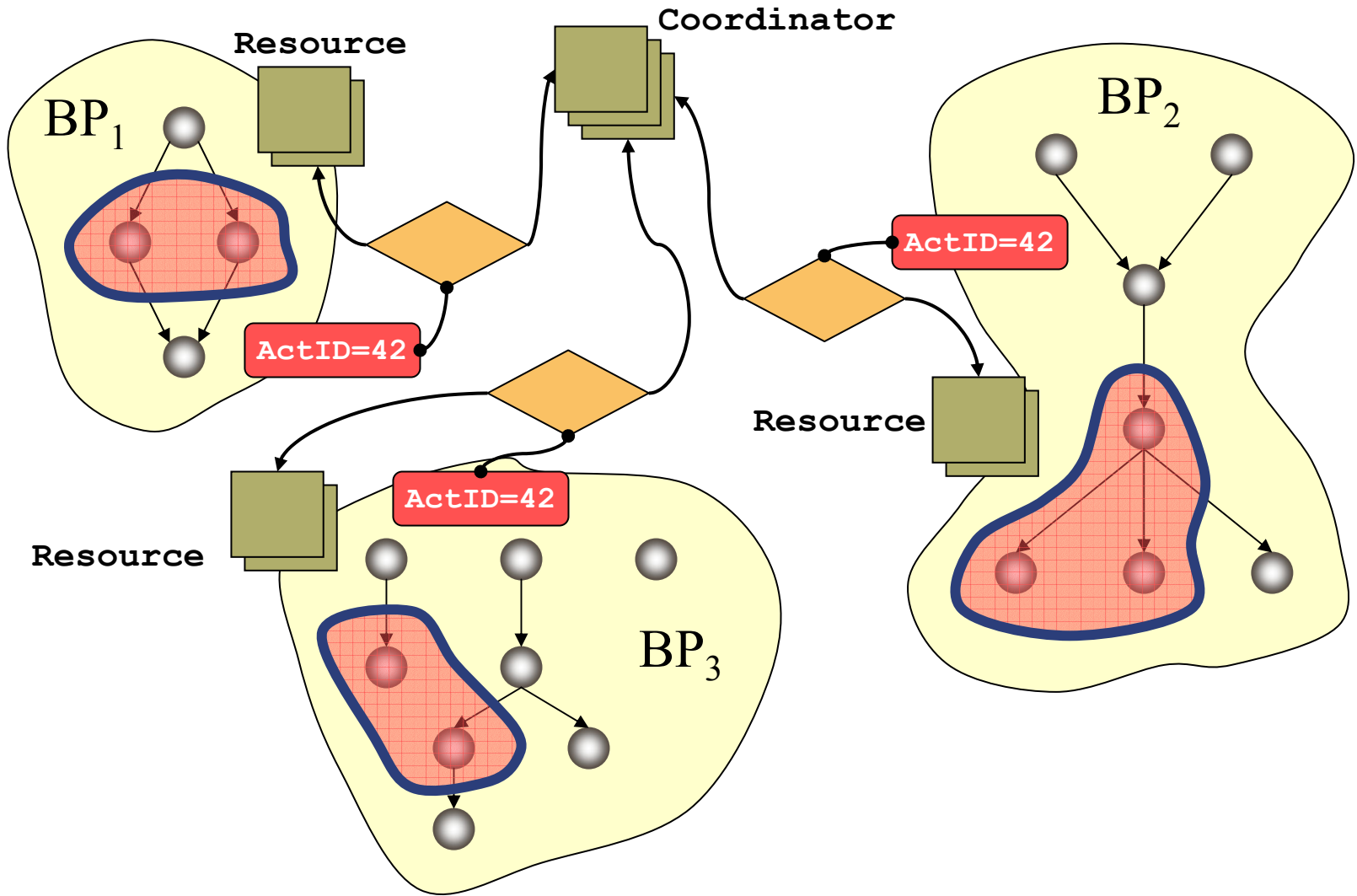
Concrete Example



LRBT Spanning Business Processes



LRBT And WS-Tx



Sample Work To-Be-Done...

To Dos

- Taxonomy of aggregation models
- Main application areas of each aggregation model
- Composability of aggregation models
 - What are “base aggregations”?
 - How to combine aggregates?

Agenda

Introduction

Virtual Components

Virtual Environments

Application Structure

Aggregations

Summary & Conclusion

Conclusion

- Huge momentum in industry on Web services
 - Standards, interoperability, implementation
- Web services are the base for the new evolving distributed computing platform
 - Virtualizing components
 - Heterogeneous “ab ovo”
- Application model is centered around...
 - SOA – Policy driven dynamics
 - Choreography – Two-level programming
- Although middleware begins to roll-out, tremendous amount of research in this space must be done

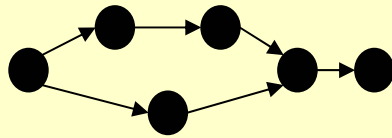
SOA Middleware Stack...



Systems Management



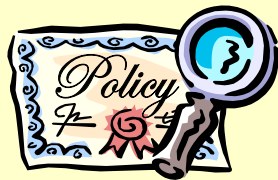
Portal



Business Processes



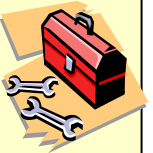
Service Aggregations



Service Bus



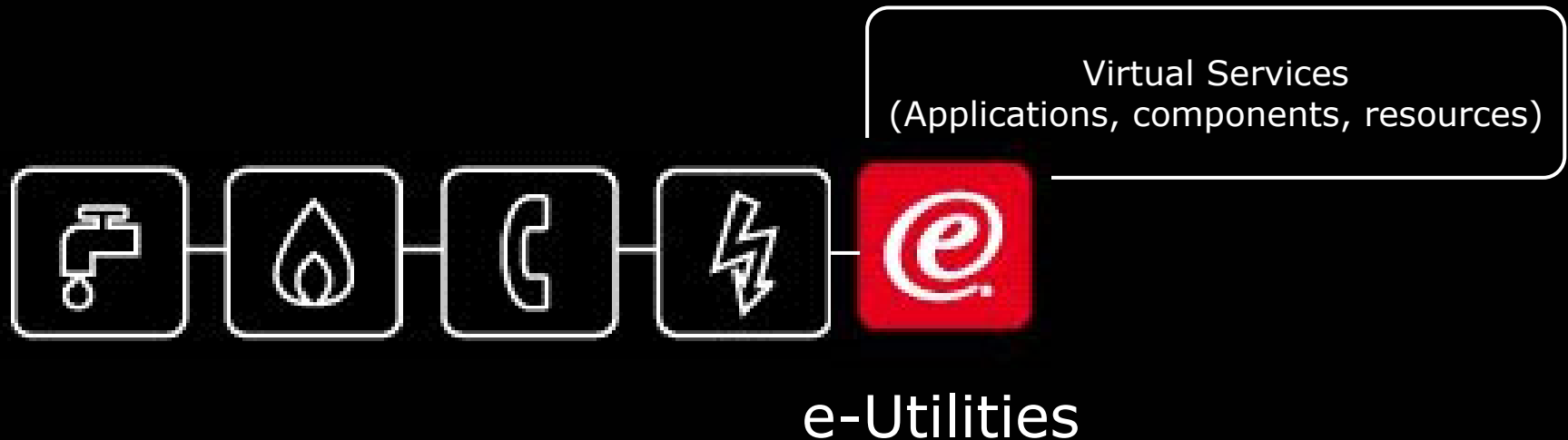
Application Server



Tools

Business Impact

Web Services → On Demand Computing



Thank You!