Interoperability in the B2B E-Commerce Era: Issues and Enabling Technologies

Dr. Boualem Benatallah

Senior Lecturer, School of Computer Science and Engineering, UNSW, Sydney, Australia

Visiting Professor, INRIA-LORIA, France

boualem@cse.unsw.edu.au

Agenda

- **Part I (Issues)**
  - **Introduction**
  - **Interoperability in B2B E-Commerce**
- **Part II (Approaches)**
  - Component-based Middleware
  - Web Services
  - B2B Interaction Standards
  - Process-based Integration
- **Summary and Outlook**

Introduction

- What is E-Commerce ?
- Information Revolution and E-Commerce
- E-Commerce Drivers
- Impacts
- E-Commerce Models
- E-Commerce Transactions

What is E-Commerce?

- Means to build efficient relationships among customers, producers, and suppliers (IEEE Communications, sept. 99)

- A set of products and services that facilitate the exchange of products, services and information over electronic networks within a company, and between companies and their customers (Gartner Group)

E-Commerce?

- **Aim**: *Conduct business transactions in a more efficient and cost-effective way*
- **Enablers**: *information and communication technologies*

  E-mail, Electronic Data Interchange Standards, e-Catalogues (e.g., Dell Computers, Amazon.com), Intranets (e.g., Cisco Connection Online), Vertical and Horizontal Portals, Data mining (e.g, Personalization), etc.

Information Revolution

- **Data**
  Unstructured (e.g., text, images)
  Semi-structured (e.g., HTML, XML)
  Tabular-data (e.g., relational databases)

- **Application**
  HTML form-based interfaces
  Gateways to DBMSs
  J2EE application servers
  Web services
  …..
- **Impacts**
  Businesses are under pressure to move their operations to the net
  Re-invention using the Internet (e.g., Ford, GM, Wal-Mart)
  Shopping on the net, banking on the net, …

E-Commerce Models

Business-to-Customer (B2C)

- Direct purchase /sale of goods and services as in retailing (Person to system)
- E-catalogue for price and product information (browsing, order placement, payment, order tracking)
- The provider defines and controls the business process

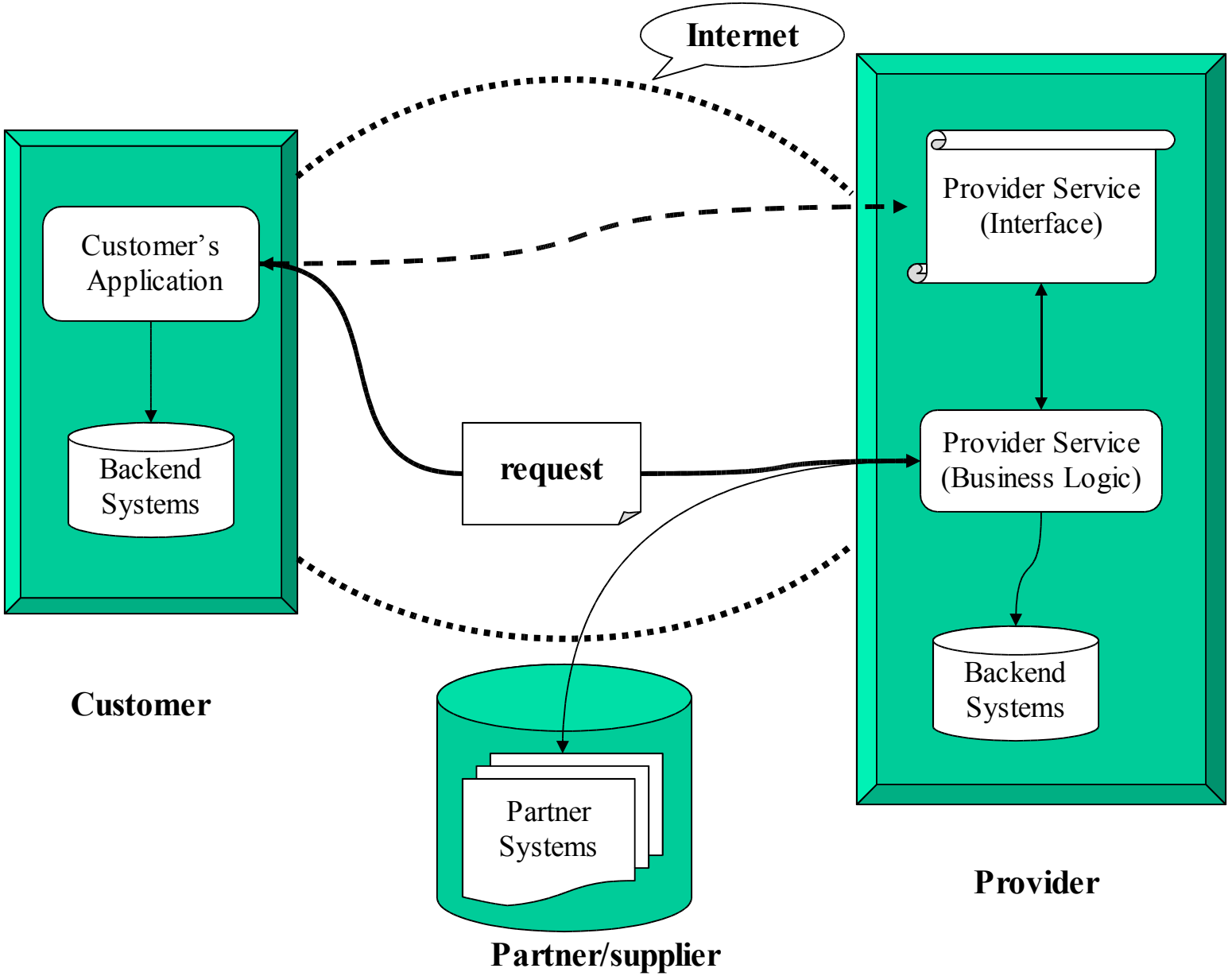
Business-to-Business (B2B)

- Interactions among customers, providers, and suppliers (multiple participants)
- Complex relationships (negotiation, static/dynamic contracting)
- Peer-to-peer collaboration to define and execute business processes, sophisticated infrastructure (e.g., workflow, EDI)

E-Commerce Models (Cont.)

- **B2C was easier to achieve compared to B2B**
- **Tutorial focuses on B2B E-Commerce**
- **B2B processes automation promise:**
  - Substantial benefits to both buyers and sellers (lower price, aggregation, lower transaction cost, better service, short procurement cycle, etc.)
  - Reduction of products and process costs
  - Make the concept of a *virtual enterprise* a reality (outsourcing to deliver greater value)
  - Fast and cost-effective *building* and *deployment* of services
  - *Customization* of existing services

B2B Transactions



B2B Applications

Example: Cisco Connection Online (CCO)

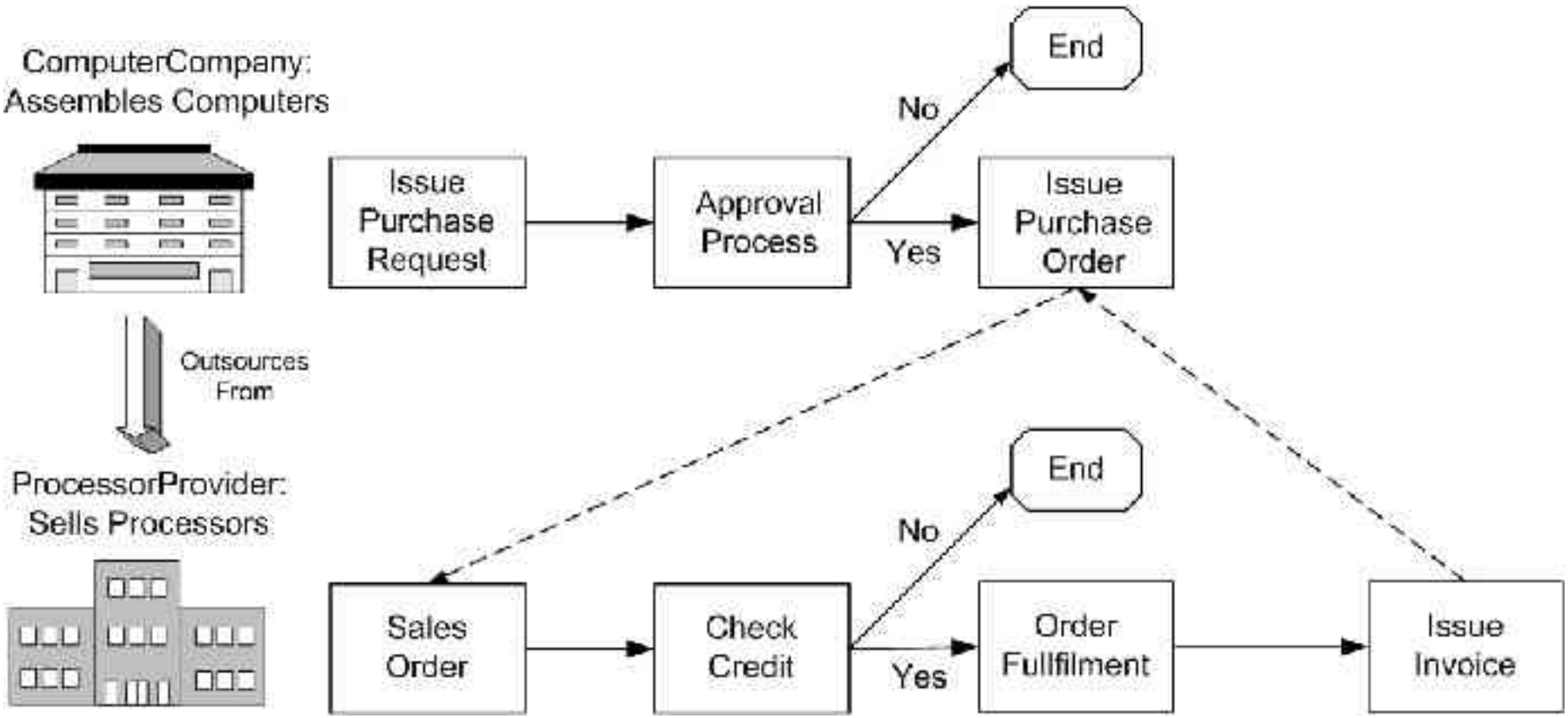
- **Front-end**
  - Pricing
  - Estimate lead times
  - Configure order
  - Sign up for a service
- **Back-end**
  - Integration with front-end order capture
  - Internal operations of order fulfillment
  - Integration with roughly 100 manufactures/suppliers

B2B Applications (Cont.)

- **Procurement**
  - Reduce costs
  - Increase efficiency (e.g., purchase of indirect goods, office supplies)
  - Customers/Buyers/Sellers (or Suppliers)

- **Value Chains**
  - Network of partners
  - Outsourcing
  - Focus on core business
  - Share costs/resources/skills
  - E.g, Computer Assembly (software / hardware parts)

B2B Application: outsourcing



Marketplaces

- Aggregation of fragmented businesses
- Vertical *vs.*. Horizontal (e.g., HealthCare Industry)
- Open *vs.* Closed
- Buyer *vs.* Seller
- Competitive *vs.* Complementary services/goods

Integration Aspects in B2B Applications

- **Both data and services (e.g., applications, workflows)**
- **Within enterprise**
  - Connect front-end and back-end systems
  - Connect legacy data sources and applications to the front-end system
  - Connect to department systems
- **Across enterprise**
  - Integration with partner systems

Integration Issues

More challenging in the Web and B2B EC Era

- Information formats are becoming more diverse
  (structured, semi-structured, unstructured)
- Information space is large and dynamic: one-to-one mappings between partner systems do not scale
- Semantic heterogeneity (both data and business processes)
- Autonomy
- Fast integration
- Across firewalls
- Evolution

Interoperability in B2B Applications

- Collaborative Applications: Coupling modes
- B2B Integration Frameworks
- Interoperability Layers
- Dimensions for Evaluating B2B Integration Solutions

Coupling modes

Centralised partnerships

- Central organization controls the global business process
- Relationships among partners: static, long-term, tightly coupled
- Focus on process efficiency
- Example
  - global customer information system several independent customer information systems developed for different purposes.

Coupling modes (Cont.)

Federated partnerships

- No central control entity
- Relationships among partners: long-term, static, loosely or tightly coupled
- Focus on process efficiency
- Example
  - A product manufacturing value chain: a participant would focus on one activity in the value chain and partners with multiple other entities in other value chain

Coupling modes (Cont.)

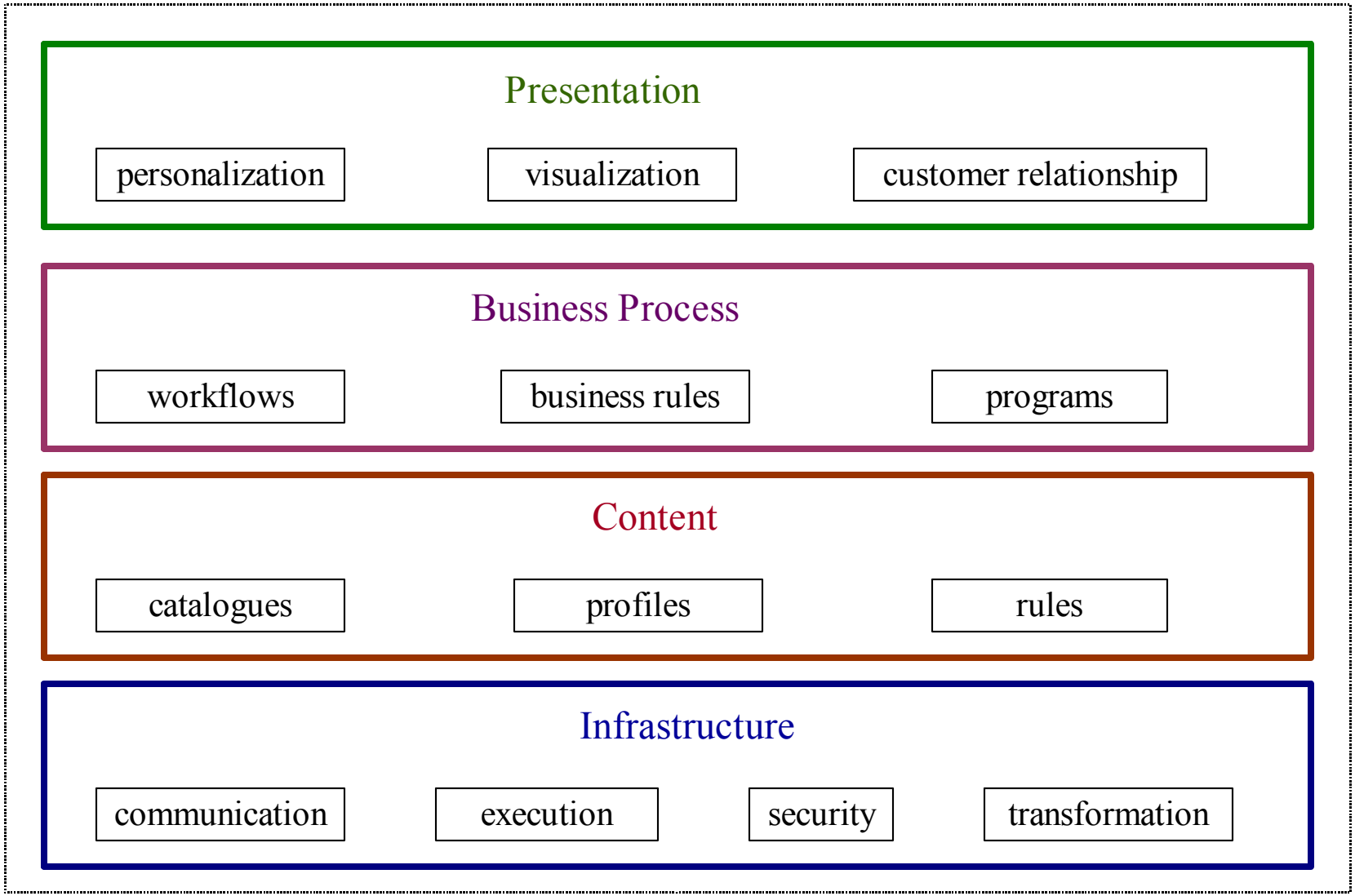
On demand partnerships

- No central control entity
- Relationships among partners: transient, loosely coupled
- Focus on transaction efficiency and value
- Fast partnership (e.g., one transaction)
- No a priori defined relationship
- Needs to dynamically discover partners
- Example
  - Online travel booking services

EC Platforms

- *Complex assembly of Web servers, databases, legacy applications, ERPs, Middleware, networking services, ...*
- **Functions**
  - Presentation of content
  - Catalogue and content management
  - Order capture and processing
  - Negotiation
  - Billing, customer support, business intelligence
  - Security
  - Integration (intra and inter enterprises)
  - …

EC Platforms (main functions)



EC Platforms: Layers and Enablers

- **Communication layer**
  - *Communication* among the participants (e.g., HTTP, FTP, VAN, publish/subscribe messaging, e.mail, event-based notification)
- **Presentation Layer**
  - Customer relationship, personalization, presentation (e.g., HTML, Java, XML/XSL)
- **Content Layer**
  - *Content and catalogue* management including storage, searching, browsing (e.g., DB, XML, HTML)
- **Business Process Layer**
  - *Collaborative activities* among participants (e.g., business rules, workflows, applications)

Presentation

- Structure, style, and display of business information
- Customisation of content and services for diverse environments (e.g., wireless devices)
  - Need to increase Buy/Browse ratio.
  - Need to give people a reason to stay longer, come back and to invite others
- E-commerce application developers need to deliver tailored experience to individuals or groups
  - The web experience can be as trivial as browsing the web site, or as significant as buying stocks.
  - Adapt the site to each user, to each visit.
  - Have interaction with users (two-way communication, e.g., feedback)
  - Understand how users are using your web sites.

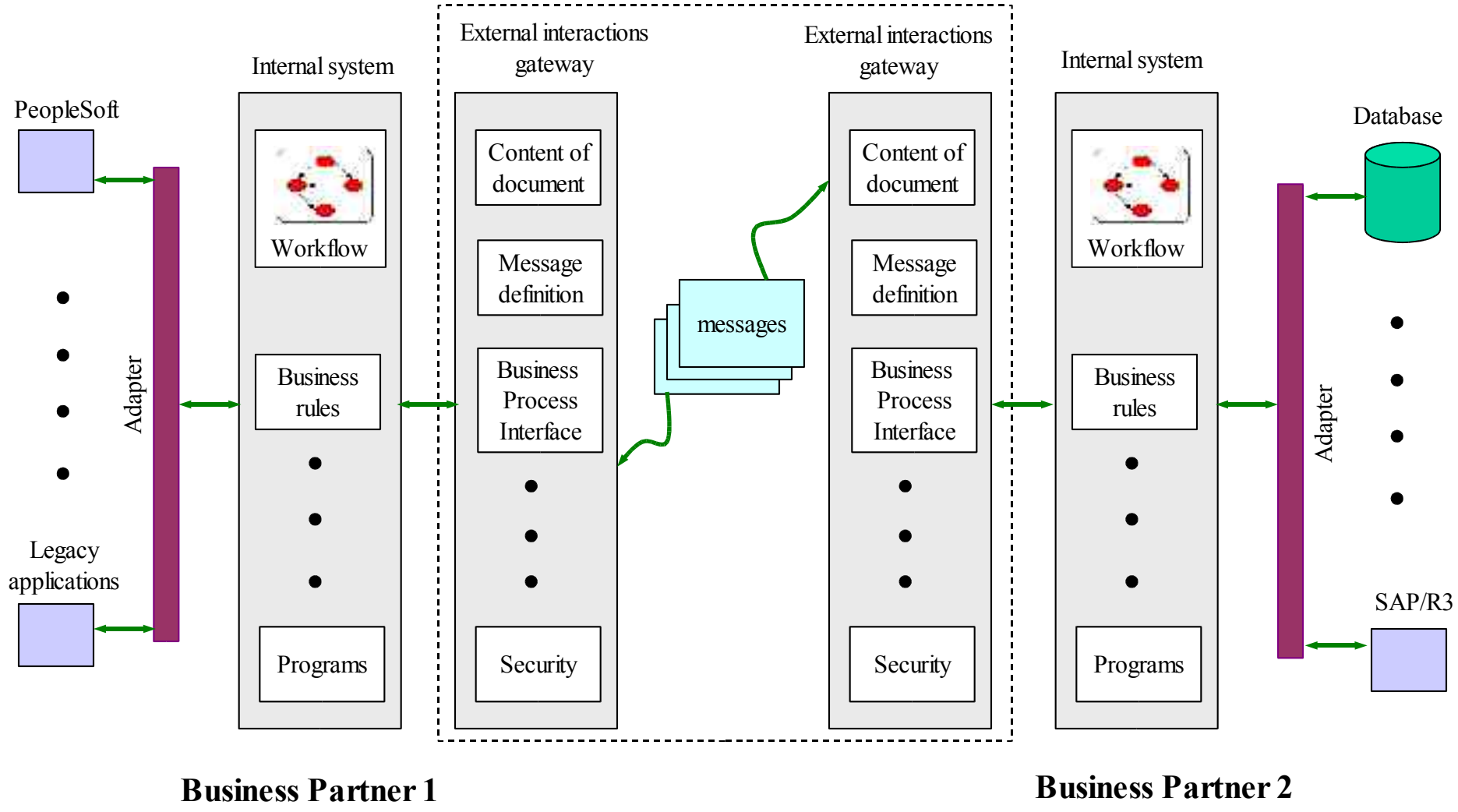
Content

- Creation and management (e.g., update/versioning) of business information (e.g., product data, transaction data, rule base, customer base)
- Models and languages to describe structure and semantics of business information
- Aggregation of catalogues
- Search and browse catalogues
- Dynamic *vs.* static content
- Caching
- HTML, text files, XML, databases
- Commercial systems: e.g., Vignette, Broadvision

Business Process

- Internal business processes, external conversational interactions among business partners
- Workflows
  - model and execute business processes, tasks, data flow, control flow, intra and inter-enterprise collaboration (e.g., shopping experience: fill form, capture form, process form, pay bill, deliver item)
- Business Rules: Event Condition Actions
  - Event: Customer A is browsing
  - Condition: A is a professor, A is a prime minister
  - Action: Display recently published books on AI,...

B2B Integration Framework



Main Functions of a B2B Integration Framework

- Content of documents: syntax (e.g., XML schemas), vocabulary (e.g., EDI messages), intent of messages (e.g., purchase order/purchase order acceptance)
- Message definition: headers (e.g., destination), communication modes (e.g., asynchronous / synchronous)
- External business processes: inter-partner collaborations
- processing of inbound and outbound messages
- Security
- Interaction protocol agreements: e.g., implementation guidelines
- Communication protocol bindings

Integration Layers

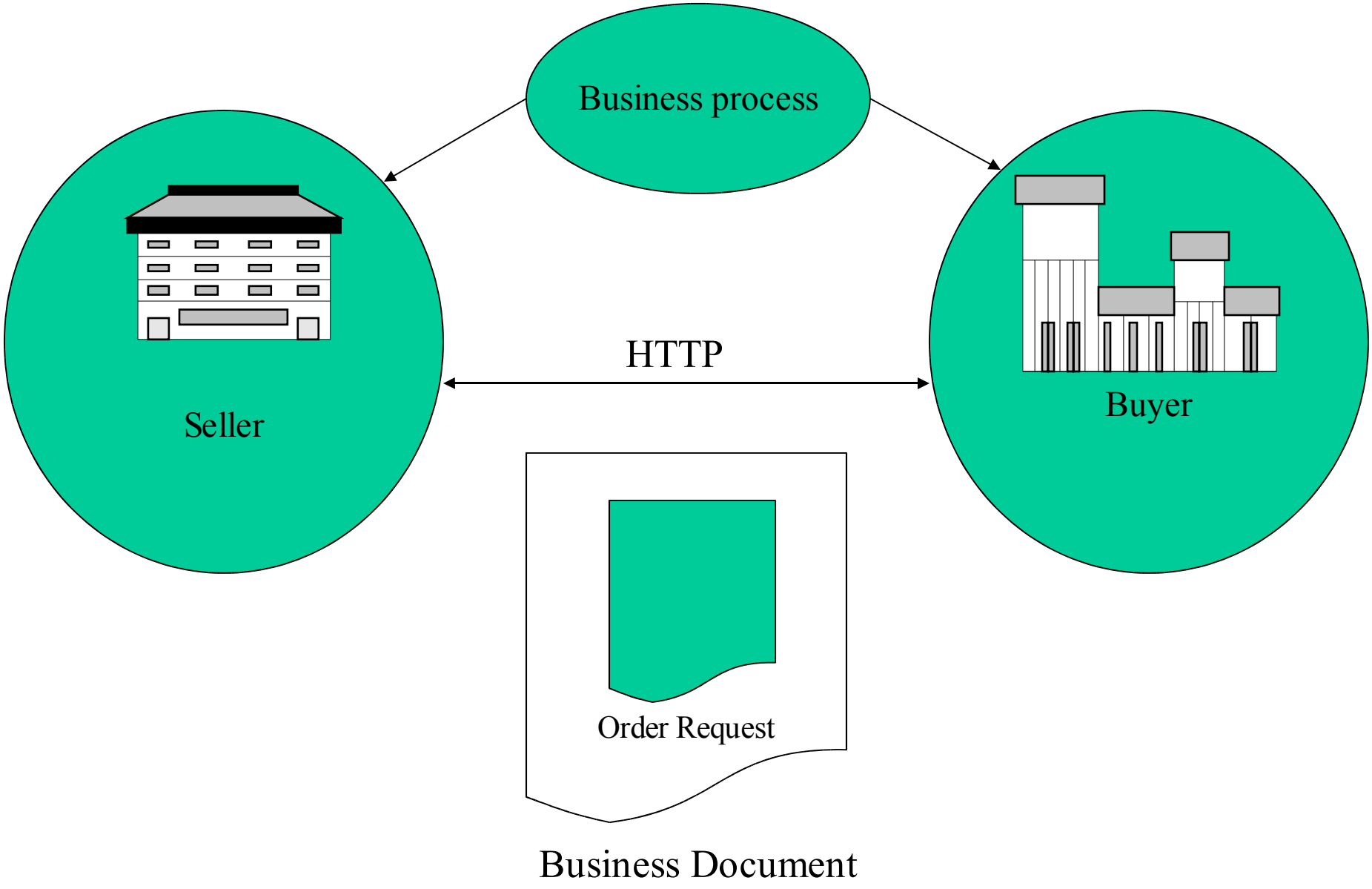
- **B2B Application**
  - Company A purchasing a product from a company B
  - Agree to collaborate, define collaborative process, and provide means to implement the collaborative process

- **Business process layer**
  - After discovering a match (e.g., using a public or a private registry), A and B need to agree on the joint business process (operations, delivery mode, contracts, etc.)

- **Content layer**
  - A needs to know and understand of the product to buy and send a purchase order to B (creation and manipulation of business information , e.g., product description, order).
  - Heterogeneity: representation/content of information

- **Communication layer**
  - There must be a way to communicate the messages that contain requests/business documents between A and B.

Interoperability layers



Communication Layer

- Exchange of messages among partners
  - Transport binding, communication modes such as asynchronous/ synchronous
  - Partners must understand messages (agree on the formats)
  - Message exchanges must be done in a secure way
  - Message exchanges must be done in a reliable manner
- Partners use different protocols (or even proprietary protocols)
  - Internet messaging (e.g., HTTP, SOAP), messaging middleware (e.g., IBM's MQSeries), EDI VANs, remote application services (Java RMI, CORBA IIOP), ...
- Interoperability objective
  - independence from transport protocols
- Interoperability solutions
  - Translate messages between heterogeneous protocols
- Examples of solutions
  - Message broker/server, message transformer (e.g., TSI soft)

Content Layer

- Issues: semantic and structural heterogeneity
- Partners must understand the structure and semantics of messages
- E.g., does a document represents a purchase order? A request for quote? A production description?
- Structure: diverse information formats, e.g., tabular-data, XML, HTML, text (e.g., different structures for a purchase order)
- Semantics: does *Price* means *Price* including *tax*?
- Partners use different models and languages to describe and organize information
- Interoperability objective
  - Independence from data models, formats, and languages
- Interoperability solutions
  - information translation and integration (reconciliation among disparate representations, vocabularies, and semantics)
  - E.g., conversion of a message from xCBL (XML common library) to cXML
  - Examples of solutions: wrappers and mediators (e.g., from OO to Rel, from XML to HTML)

Business Process Layer

- Semantics of interactions (joint business process)
- Partners must agree on the choreography of interactions and meaning of messages
- E.g, steps (send order, process order, deliver product), deals (a purchase is refundable after 2 days)
- Semantics of interactions must be well defined, such that there is no ambiguity as to:
  - What a message may mean? What actions are allowed? What responses are expected?
- For example, if a company A requires an acknowledgement of purchase orders from its partners, then partner processes must have a corresponding activity
- Support for monitoring and enforcing agreements must be provided
- Advertisement and discovery of terms and capabilities using registries
- Interoperability objective: allow autonomous partners to advertise their terms and capabilities, and engage in peer-to-peer interactions with any other partners.

Business Process Layer (Cont.)

- Component-based solutions
  - Messaging Middleware and DB technology
  - Abstract interfaces, remote operation invocation, connector for back-end systems, connection and coordination of operations
  - Business processes are worked out offline
  - Examples: CrossWorlds, CORBA-based solutions
- Document-based solutions
  - Interaction = a set of documents following a protocol
  - No prior agreement, partners publish their documents independently, self-describing
  - Examples: EDI, RosettaNet
- Process-based solutions
  - Support description of business process directly
  - Several emerging standards propose solutions in this direction (e.g., BPEL4WS, ebXML BPSS)

Evaluation of B2B Integration Solutions

- Several solutions exist
- Types of interactions depend on usage scenarios, parties involved, and business requirements.
- It is important to understand requirements and the related tradeoffs
- Quantitative evaluation?
- Informal but useful guidelines for assessing integration solutions = B2B Integration (B2Bi) dimensions
- We will discuss few dimensions: ***coupling among partners, Heterogeneity, Autonomy, Adaptability, External Manageability, Security, Scalability***.
- Existing solutions: variation in their tradeoffs with regard to B2Bi dimensions

B2Bi Dimensions: Coupling among partners and scalability

- *Coupling among partners*
  - *Degree of tightness*
  - Lifetime of relationships: long term, short term (e.g., one transaction)
  - Partnership mode: centralised, federated, on-demand
  - Process *vs.* transaction efficiency and value
- Scalability
  - *Ability to grow in one or more dimensions such as volume of data, number of transactions, number of relationships (transparent behaviour)*
  - Support of new functionality, merge with other organizations
  - Relationships with partners: how many?, what types?
  - Cost and effort to support new relationship is an important indicator for scalability

B2Bi Dimensions: Heterogeneity

- Data heterogeneity
  - Structure: disparate data representations, common layer: open, no-proprietary standards (e.g., XML)
  - Semantics: standardised vocabularies for different industry sectors,
- Process heterogeneity
  - Semantics of interactions
  - Global business process: APIs, exchange of business documents, inter-enterprise workflows
  - Partners may use different strategies for conducting business

B2Bi Dimensions: Autonomy

- *Degree of compliance of a partner to global control rules*
- Partner systems may be autonomous in their design, communication, and execution
- Partners select process, content, and communication models, languages, and protocols
- Autonomy may impact the complexity of integration solutions, degree of interoperability, flexibility of local control, etc.
- Full autonomy: a partner = black box, flexibility of change, difficult to achieve (may require sophisticated translation capabilities), minimal interoperability
- 0 autonomy: reveal all internal information (local processes), may be unacceptable

B2Bi Dimensions: Adaptability

- *Degree to which a partner is able to adapt to changes (also called agility)*
- Both operational (e.g., server load) and market (e.g., user requirements) environments are not predictable
- Changes are need to support:
  - new requirements, new technologies, new policies
  - Customisation and upgrade
- Changes are more frequent at the content and business process levels
- Changes may require propagation to internal and external systems
- Impact of changes depends on the degree of coupling

B2Bi Dimensions: External Manageability

- *Degree to which a partner is to be visible and manageable by external partners*
- Facilitates process monitoring and control
- Tracks changes
- Allows Interaction during service provisioning
- Requires to expose sufficient information (e.g., measurements, control points)
- Real time detection and corrections of deviations become possible
- May require complex descriptions (may be justified if it provides quality of service for e.g), impact autonomy!

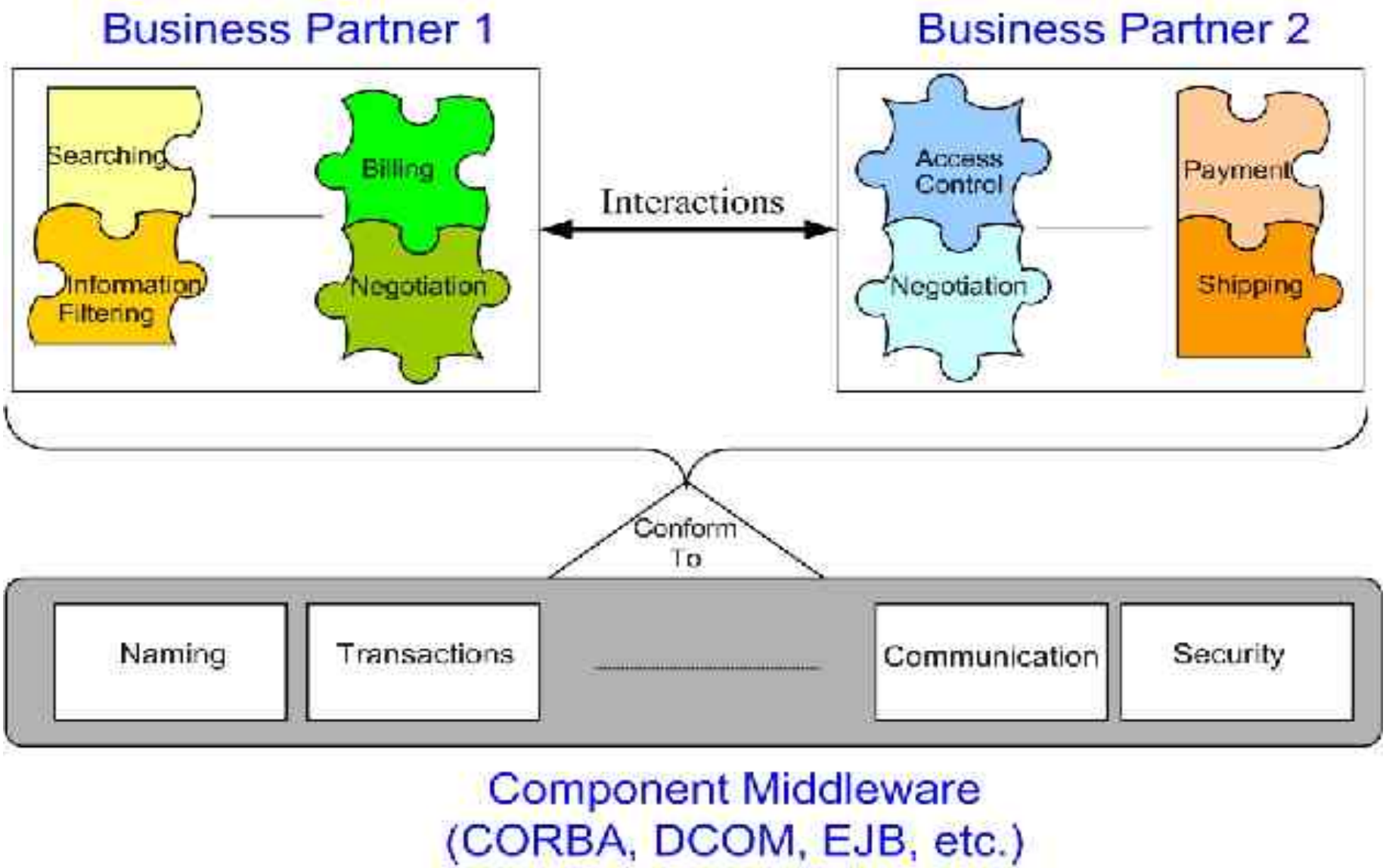
B2Bi Dimensions: Security

- EC applications may need to cross corporate firewall and security systems
- Security is a major concern for inter-enterprise interactions
- Authentication and access auditing
- Communication integrity
- Confidentiality
- Non-repudiation
- Sophisticated security measures must be in place to give customers and partners the confidence that their transactions are safely handled
- May require higher initial deployment and maintenance costs in remote calls based systems
- Interactions may be based on limited trust between partners, little knowledge of partners, transient collaborative agreements, etc.
- Shared information may include only limited capabilities of services

PART II: Integration Approaches

- **Component-based Middleware**
- Web Services
- B2B Integration Standards
- Process-based Integration

Using Component-based Middleware



Component-based Middleware (Cont.)

- Typically rely on distributed object frameworks such as CORBA, DCOM, EJB and other state of the art technologies such as database gateways and transaction monitors
- Separation between applications and infrastructure services (e.g., persistence management, security management, transaction management, trading, event, naming services)
- Platform and language independence
- A related approach is Enterprise Application Integration (EAI) suites (e.g., IBM's MQSeries, TibcoSoftware's TIB/Active Enterprise Suite, TSI's Software's Mercator product, and IBM SanFrancisco)
- EAI suites provide standard data and application integration facilities (e.g., application adapters, data transformations, and messaging services)
- Some EAI suites provide messaging services among different ERP systems, e.g., TSI Software's Mercator offers messaging services between SAP R/3 and PeopleSoft

Component-based Middleware (Cont.)

- Components represent high-level services such as business objects (e.g., purchase order placement, payment)
- Developers focus on component specification (e.g., using CORBA IDL), they do not need to know where objects are located, in which languages they are implemented, how they communicate, etc.
- Integration is based developing unified interfaces to access heterogeneous and distributed systems
- Emphasis more on the syntactical integration: wrapping heterogeneous systems, routing requests, remote operation invocation
- API based Integration: business objects are wrapped with explicit interfaces, they communicate by making remote calls directly to their peers
- Tight coupling between partner systems (operation invocation)

Component-based Middleware: Interoperability layers

- Communication layer: CORBA IIOP, Java RMI, RPC, etc. In general, communication is synchronous
- Typically an OO model is used to describe service interfaces (input parameters, out parameters, operation names)
- Business processes are worked out offline
- Sometimes pre-defined components that provide basic business application functionality exist
- Interfaces do not capture business process semantics beyond enumerating interface types

Component-based Middleware: B2Bi dimensions

- They generally assume a tight coupling model
- Creation of a relationship with a partner application (in CORBA): define IDL interfaces, generate stub and skeletons, implement the service and publish it
- Security : May require higher initial deployment (e.g, access rights), security support provided by the infrastructure (e.g., CORBA security service)
- Because of tight coupling, changes to back-end systems, mediator framework, and business applications must be coordinated across all the components
- Separation between interfaces and implementations (autonomy)
- Appropriate to integrate small number of tightly coupled services

PART II: Integration Approaches

- Component-based Middleware
- **Web Services**
- B2B Integration Standards
- Process-based Integration

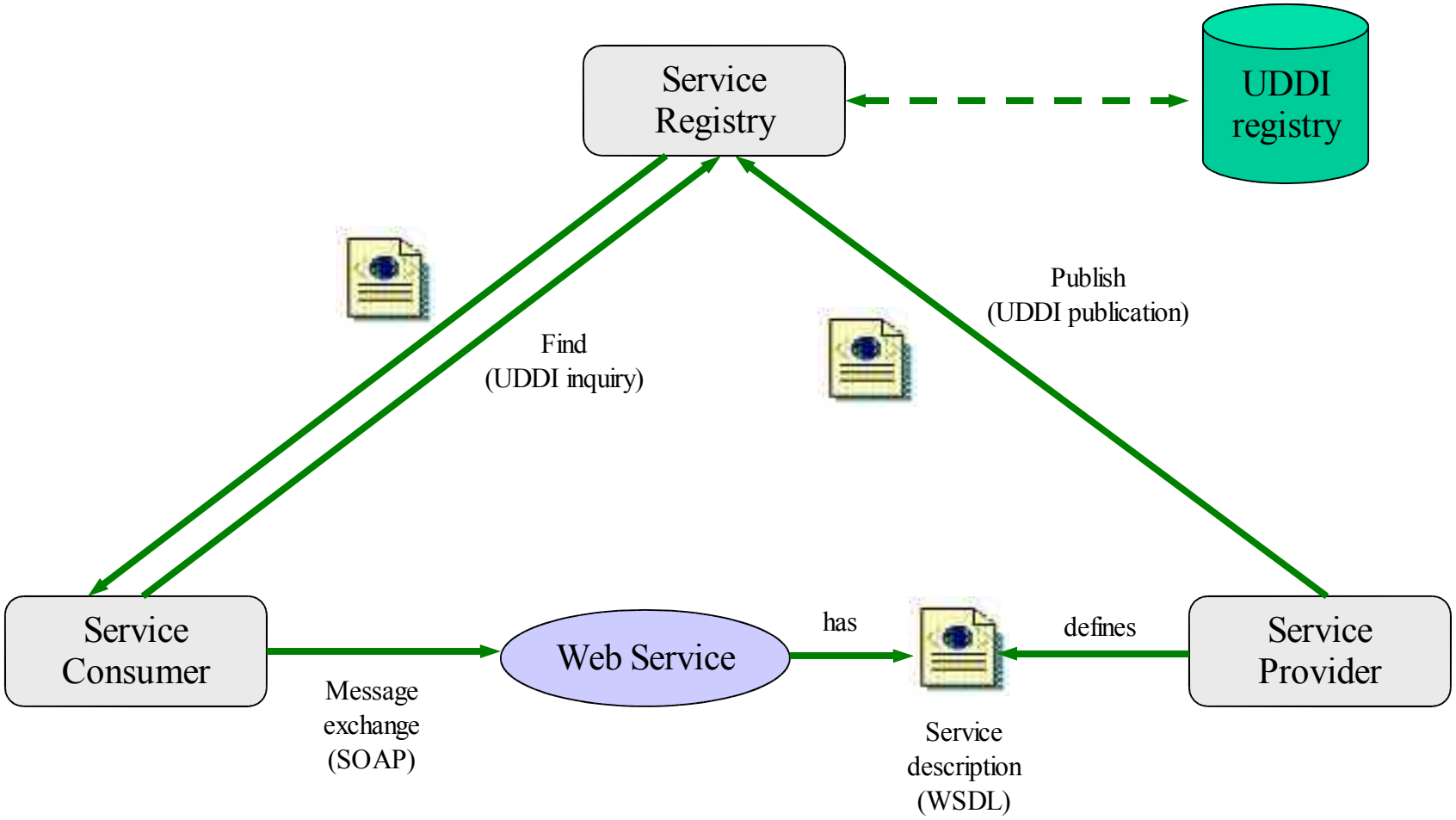
Web Services

- Applications accessible via programmatic means
- Different types: Information delivery (e.g., stock quotes), transactional services (e.g., hotel reservations), supply chain
- Web services are emerging as a middleware technology for *loosely coupled integration*: document-based integration
- Build upon XML technologies
- Enjoy support from major industry players including IBM, Microsoft, SUN, BEA
- Several ongoing standardisation efforts (e.g., SOAP, WSDL, UDDI), but still lack support for important infrastructure services as security, transaction, and event management services.

Web Service Infrastructure Stack

Collaboration **ebXML (electronic business XML)** **BPEL4WS(Business Process Execution Language for Web Services)**
Advertisement and Discovery **UDDI (Universal Description, Discovery, Integration)**
Description **WSDL (Web Service Description Language)**
Document Exchanges **SOAP (Simple Object Access Protocol)**
Internet Protocols (HTTP, TCP/IP)

Web Services: Reference Architecture



Document Exchanges using SOAP

- XML-based protocol for exchanging messages across the Internet
- Relies on Internet transport protocols such HTTP
- Types of messages : Request (e.g., invoke a service operation) and Response (e.g., results of a service invocation)
- SOAP message (envelope) = header + body
- Header: entries to specify intended purpose (e.g., service invocation, invocation results), reliability, sender credentials, recipients, etc.
- Body: request message (operation name, values of input parameters), response (results of service invocation)
- SOAP implementations exist for several programming languages (e.g, Java, C): translation of SOAP messages to/from service business logic (e.g, Java class)

SOAP: an example of a request

```
POST /carRent HTTP/1.1
Host: www.axac.com
Content-Type: text/xml; charset="utf-8"
Content-Length: 127
SOAPAction: "http://www.anywhere.com/rentCar"
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope"
   SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
   <SOAP-ENV:body>
      <m:rentCar xmlns:m="http://www.anywhere.com/rentCar">
           <customer> Arun Sharma </customer>
           <rentalDate>18/05/2002</rentalDate>
           <returnDate>20/05/2002</returnDate>
      </m:rentCar>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP: an example of a reply

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: 234
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope"
    SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
    <SOAP-ENV:Body>
        <m:RentcarResponse xmlns:m="http://www.anywhere.com/rentCar">
            <rentalFees>234.00</rentalFees>
        </m:RentcarResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Service Description using WSDL

- An XML-based language for describing services
- Service description = collection of end points.
- End-point (interface) : abstract definition + implementation binding
- Abstract definition : types of messages exchanged, operation signatures
- Implementation binding: communication protocol to use, location of service, how an interaction occurs over a given protocol

Abstract Description

- Data exchanged (e.g., input or output data)
  - **message:** message name + parts, e.g., **cardRentInput**
  - **part:** e.g., **custome**r name and **creditcardnumber** are parts of **cardRentInput** (typed according XML Schema for instance)
- Operation description
  - A message exchange pattern: one way, request-response
  - Name (e.g., **RentCar**) + input message (e.g., **cardRentInput)** + output message (e.g., **CartRentOutput**) …
- portType = a set of operations supported by an end point

Abstract description: An example

```
<definitions name="carRent" ....>
   <types>
      <schema targetnamespace="http://example.com/carRent.xsd"
            xmlns="http://www.w3.org/2000/10/XMLSchema">
         <element name="Customer">
            <complexType>
               <all><element name="Name" type="string"/>
                     <element name="CreditCardNo" type="string"/>
               </all>
            </complexType>
         </element>
      </schema>
   </types>
```

Abstract description: An example (cont.)

```
….
<message name="carRentInput">
      <part name="customer”, element="tns:Customer"/>
      <part name="rentalDate" type="xsd:date"/>
      <part name="returnDate" type="xsd:date"/>
 </message>
 <message name="carRentOutput">
       <part name="rentalFee" type="xsd:float"/>
</message>
<portType name="carRentPortType">
      <operation name="RentCar">
         <input message="tns:carRentInput"/>
         <output message="tns:carRentOutput"/>
      </operation>
</portType>
```

Implementation Binding

- Mapping between abstract operations and concrete service implementations
- **binding:** how interactions (portType) occur over a message exchange protocol (e.g., map **rentCar** operation to a **SOAP-based concrete operation**)
- **port**: a network address where to locate a binding
- **service** : a set of ports

Implementation Binding: An example

```
<binding name="carRentSoapBinding" type="tns:carRentPortType">
    <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="rentCar">
      <soap:operation
soapAction="http://www.anywhere.com/rentCar"/>
      <input><soap:body use="literal"/></input>
      <output><soap:body use="literal"/></output>
    </operation>
</binding>
…
<service name="CarRentService">
    <port name="carRentPort" binding="tns:carRentSoapBinding">
        <soap:address location="http://example.com/carRent"/> </port>

    ...
</service>
```

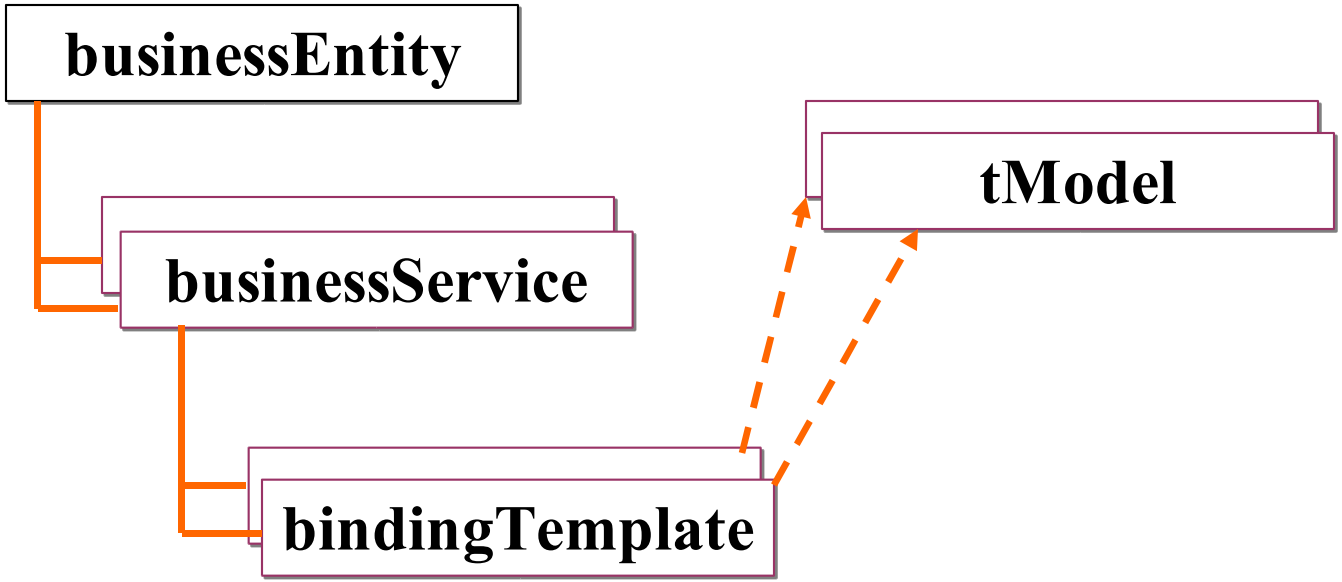
Advertising and Discovering Services using UDDI

- Service directory: efficient discovery of services
- Content of a directory
  - Meta-data about services (e.g., categories of services, service providers)
  - Access information (e.g., location, interface, implementation bindings)
- UDDI provides advertisement and discovery APIs
- UDDI service directory is like the phone directory for Web services
- UDDI directory data structures: **white pages, yellow pages, and green pages**

UDDI Directory Information

- White pages:
  - Business name
  - Text description
  - Contact info (e.g., names, phone numbers, fax numbers)
  - Known Identifiers (e.g., according to known classification)
- Yellow pages: *Business* and *service categories*
  - Industry: NAICS (North American Industry Classification System)
  - Product/Services: UNSPSC (Universal Standard Products and services Code System)
  - Location: Geographical taxonomy (ISO 3166)
- Green pages: access information
  - Service descriptions
  - Binding information

UDDI Directory Information



businessEntity

- Information about a provider and its services
- Business Key (UDDI specific, generated when during registration)
- Name
- Description (e.g., a text)
- Contacts (e.g., phone, address, email)
- Business services
- Identifier bag (e.g., identifiers that a business may known by)
- Category bag (e.g., NAICS code)
- ….

An example of business registration

```
<businessEntity businessKey="089B5-ER8-AC09-599CF7">
    <name>Anywhere Ltd</name>
    <description xml:lang="en"> cars for rent</description>
    <businessService businessKey=" 089B5-ER8-AC09-599CF7"
                serviceKey="12FF-2AF3-45FB-09AF7">
                <name>carRental</name>
          </serviceInfo>
          ……..
    </serviceInfos>
```

businessService

- Information about a specific service
- Service Key (UDDI specific, generated when during registration)
- Business Key (A reference to the provider, a business entity)
- Name
- Description (e.g., a text)
- Binding templates: service access information
- Category bag (e.g., UNSPSC code)
- ….

bindingTemplate

- Represents a service end point (may be several end points for one service)
- Access point (Address of the service, e.g., URL, e.mail, phone number!)
- Binding key
- Service key
- Description
- tModel Instance Details: how an interaction occurs with the service (green pages information)
- ….

tModel

- Contains reference to a technical specification (e.g., a WSDL document, RMI Remote Interface, CORBA IDL)
- A tModel is defined and registered independently of services
- Services make references to existing or newly created tModels
- A tModel is described by its key, name, description, identifier, category, and overview document
- A tModel may specify a category to which it belongs (this will facilitates discovery of tModels and linking them to service descriptions)
- In fact classification systems (I.e, NAICS, UNSPSC, ISO 31 66) are registered as a tModels

Registering WSDL service specification as tModel in UDDI

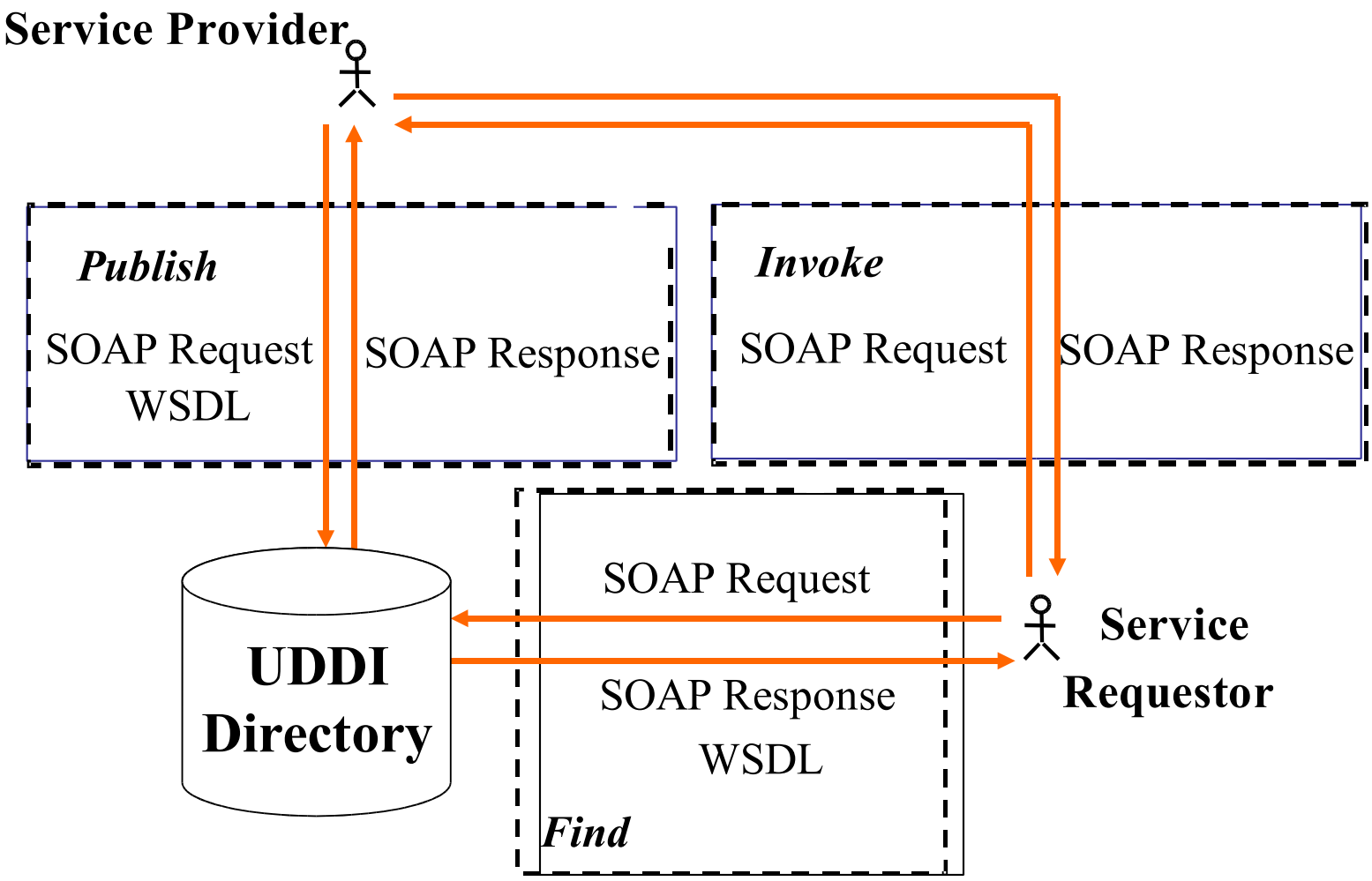
- Allows the use of UDDI to search for Web services which are described using WSDL
- The element **overviewURL** of the element **overviewDoc** of the tModel refers to the WSDL document that describes a service
- The element **categoryBag** of the tModel refers to the classification **wsdlSpec** of **uddi-org:types** taxonomy
- Once the tModel exist in UDDI, a **businessService** can refer to it in its **bindingTemplate**

UDDI APIs - SOAP messages

- **Inquiry API**
  - find_business
  - find_service
  - find_binding
  - find_tModel
  - get_businessDetail
  - get_serviceDetail
  - get_bindingDetail
  - get_tModelDetail
  - ...

- **Publisher API**
  - save_business
  - save_service
  - save_binding
  - save_tModel
  - delete_business
  - delete_service
  - delete_binding
  - delete_tModel
  - ...

Integrating UDDI with WSDL and SOAP



Web Services (Cont.)

- *Service oriented paradigm promises to allow autonomous and heterogeneous partners to come online, advertise their terms and capabilities, and engage in peer-to-peer interactions with any other partners.*
- Self-described and loosely coupled
- Can be published, discovered, and invoked over the Internet
- Provide standardization for enterprise application integration (build upon XML standards)
- Interoperability: communication layer, content layer (addresses structure heterogeneity via XML)

Critical Issue: interoperability at the business process layer

- Interoperability at the business process layer requires the understanding of the behavior of partner public processes (external conversations).
- Traditional EAI middleware: component interface describes very little semantics (e.g., no public description of transactional semantics), business process is usually agreed upon off-line.
- Web services: everything should be in the service description (self-describing!), essential in dynamic and large environments
- Automation requires rich description models but a balance between expression power and simplicity is important for the success of the technology

An approach to describe conversation semantics

Identify and define service conversation abstractions such as:

- Transactional semantics
- Temporal constraints

To provide support, e.g., for:

- Clients in searching services based on these properties
- Providers in automating the enforcement of the properties

Identifying Abstractions of Web Service Conversations

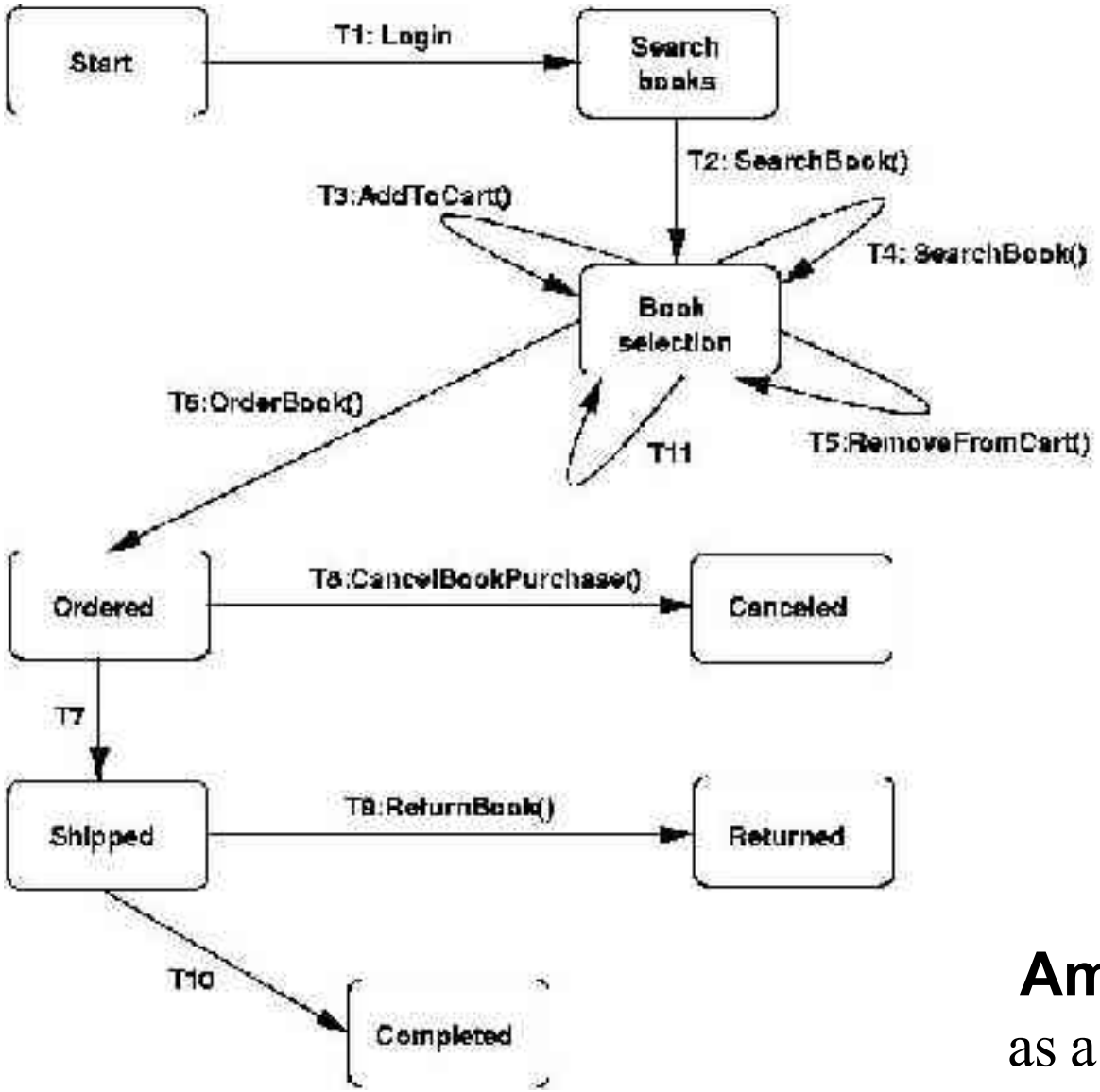
- Analyzing Web portals rather than Web services since:
  - Developers are still fighting with basic interoperability of Web services (SOAP connections)
  - Web-based e-commerce is a mature area: portals often include ``terms and conditions'' documents.

- Analysis of about 20 Web portals including **Amazon.com**, **Travelocity.com**, and **Expedia.com**

Web Portal Analysis

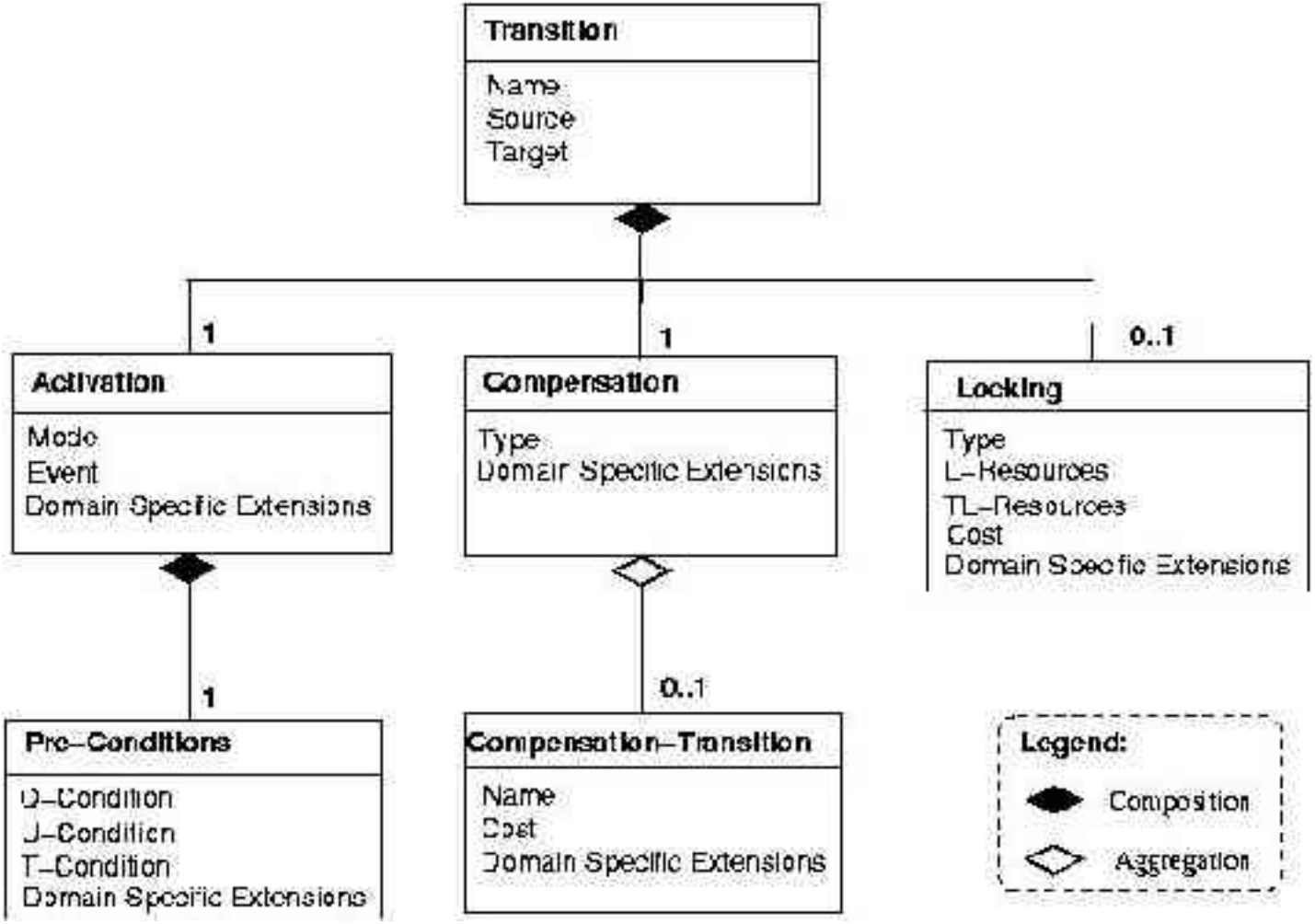
WEB PORTAL	SERVICE	BEGIN	COMMIT	ROLLBACK	LOCK/HOLD	COMPENSATE
Amazon.com	Sell Books	At checkout time when customers finalize the purchase of one or more items	At book shipping time	Cancel via phone, web, or email before product shipped	None	Return within 30 days for refund if in good condition (or in ANY condition for books recommended by Amazon). MANUAL PROCESS
Travelocity.com	Rent cars	At checkout	At end of checkout process	N/A	None	Rental may be compensated within a time T from the pick up time, otherwise a fee F needs to be paid. T and F vary by company, car size, agreed rate, etc.
Expedia.com	Sell flight tickets	At checkout	At end of checkout process	Session expires (no explicit cancel required) before ticket issued	Locks seats on hold until 12am next day (available only for certain flights). Does not guarantee fare	Return: depends on agreement with airlines, fare selected, etc. It is therefore on a case-by-case basis. E-PROCESS

Embryonic Conversation Model

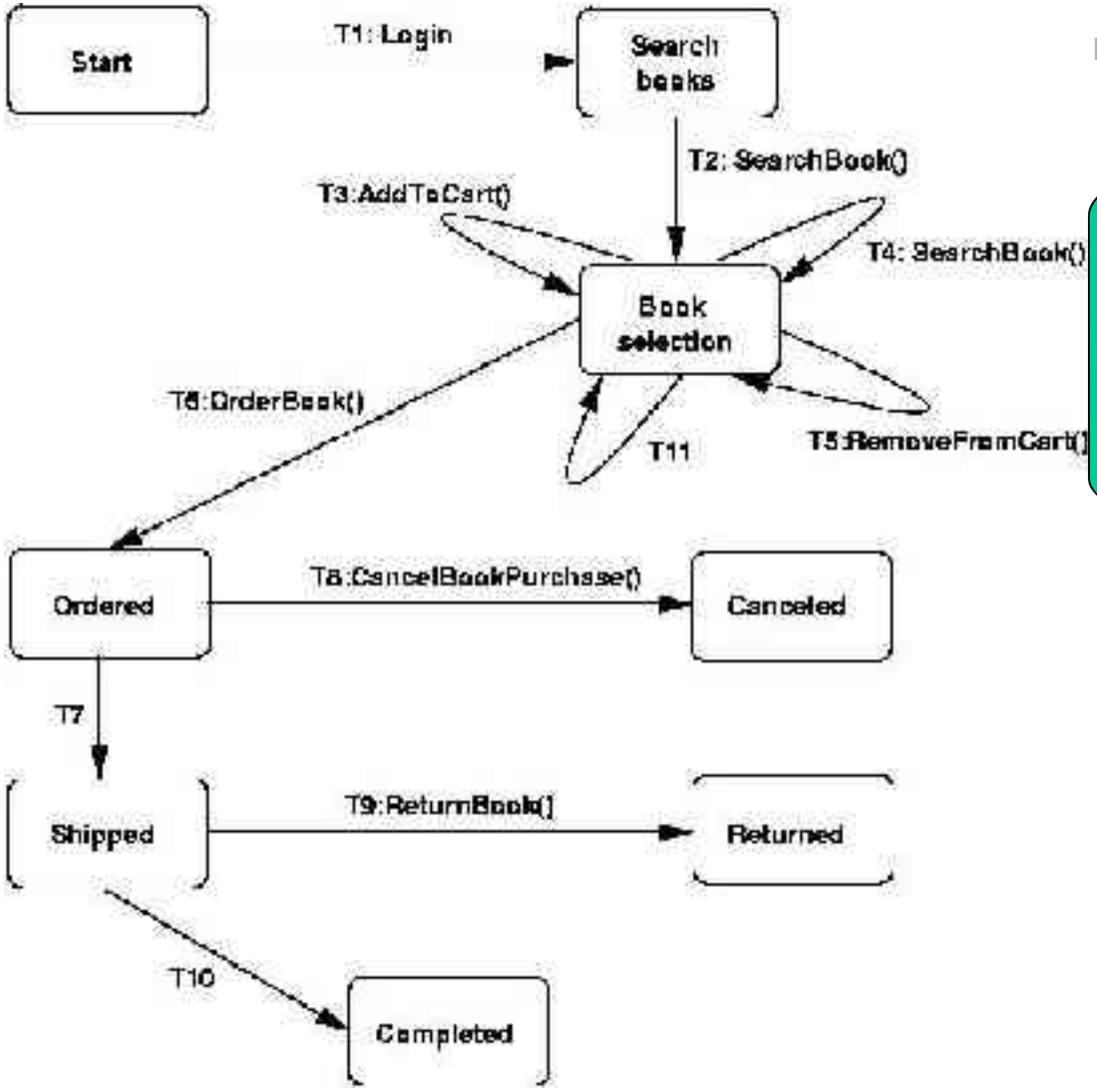


Amazon.com
as a Web service

Identified Behavioral Properties



Activation Property (cont'd)



- **Examples**:

mode := "user"
event := "ReturnBook"
O-condition := True
U-condition := membership = 'gold'
T-condition := "within 30 days after end(T7)"

mode := "provider"
event := "30 days after end(T7)"
O-condition := True
U-condition := True
T-condition := True

Transaction Property

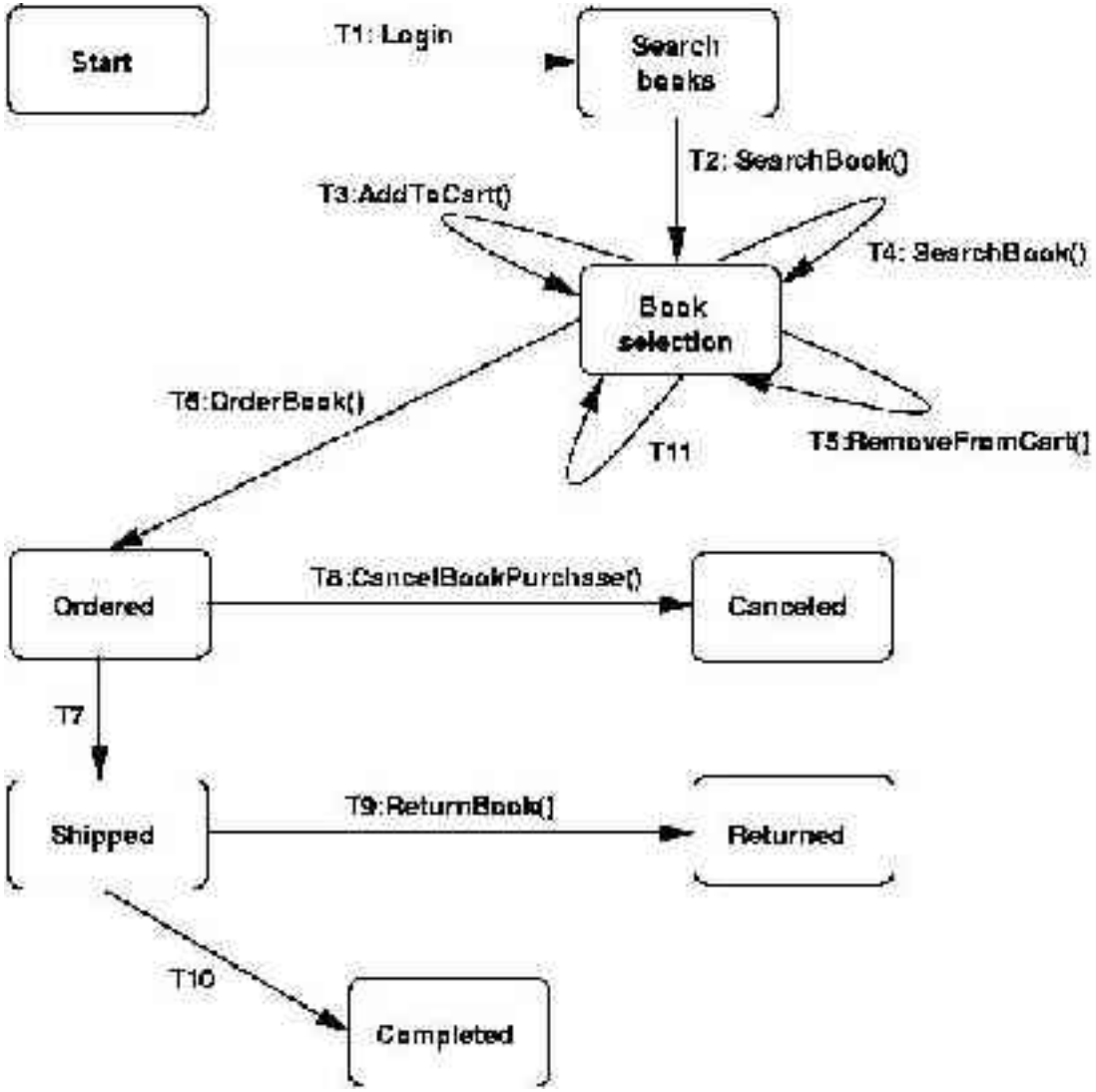
- Specifies the effect of a transactional transition on the requester side:
  - **Effect-less**: no effect on the requester
  - **Credential-disclosure**: the requester is required to reveal certain credentials but no effect on the requester
  - **Definite**: the effect is permanent and cannot be compensated
  - **Compensatable**: some effect on the requester that can be undone
  - ….

Compensation Property (cont'd)

- **Examples**:

name := "T7"
type := "Compensatable"
compensation-transition := "T9"
cost := 10% * books_price

name := "T10"
type := "Definite"



Resource Locking Property

- Specifies temporary reservation of service provider resources for a requester when invoking a transition
- Two types of resource locking:
  - **Lock** (L)
  - **Tentative-lock** (TL)

Using Abstractions

- **Service discovery** (e.g., require that a selected service support operation cancellation within a time interval)
- **Validation of service composition models** (e.g., if op is compensatable by c-op, composition logic must include support of receiving op-c after op has completed)
- **Joint analysis of compositions and conversations** (e.g., composition **sequence (op1, op2),** if **op1 or op1 (definite),** definite, then the composition behavior will **not be atomic)**.
- **Generation of conversation models** (e.g., op= sequence (op1, op2), op1/op2 are compensatable, op is compensatable via sequence (c-op1, c-op2)).
- …

PART II: Integration Approaches

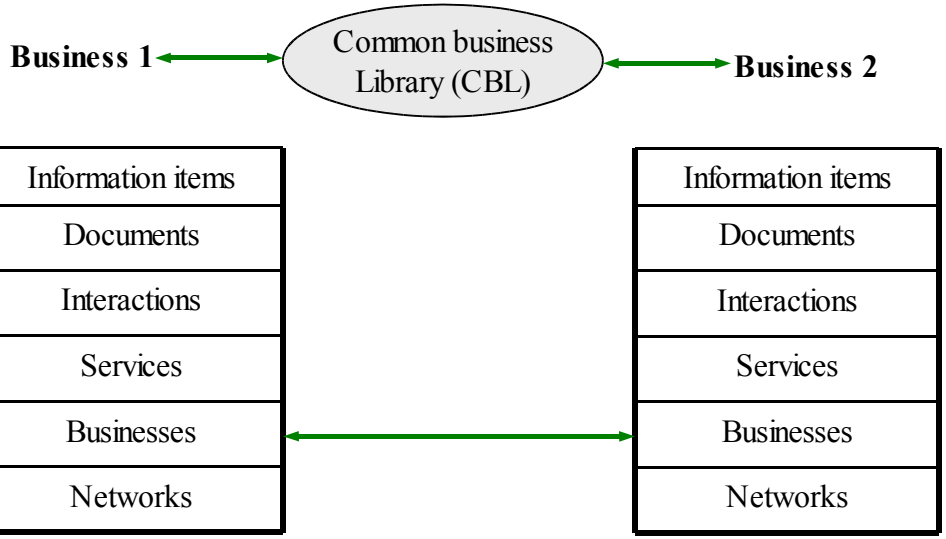
- Component-based Middleware
- Web Services
- **B2B Integration Standards**
- Process-based Integration

B2B Interaction Standards

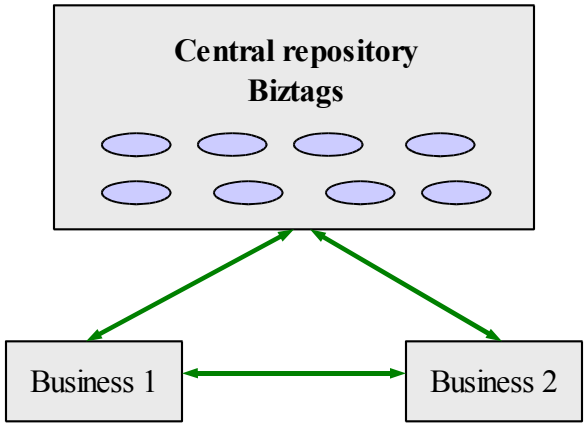
- Definitions for documents and conversational interactions among partners
- Formats of message envelope and related security aspects
- EDI (Electronic Data Interchange) standards: ANSI X12 and UN/EDIFACT
- Several XML-based standards exist including:
  - eCO (Initiative of CommerceNet)
  - Commerce XML (cXML) (Ariba)
  - RosettaNet (RosettaNet.org)

  …

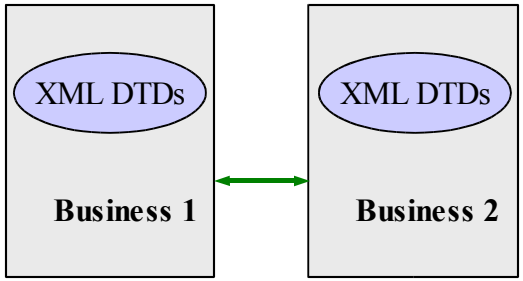
B2B Interaction Standards



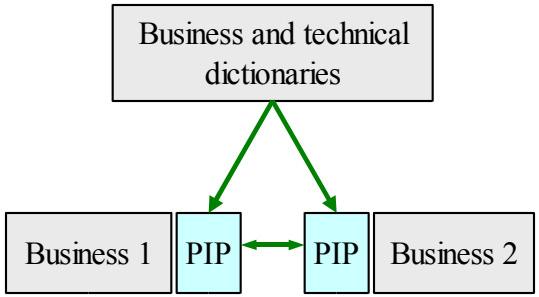
1. eCO



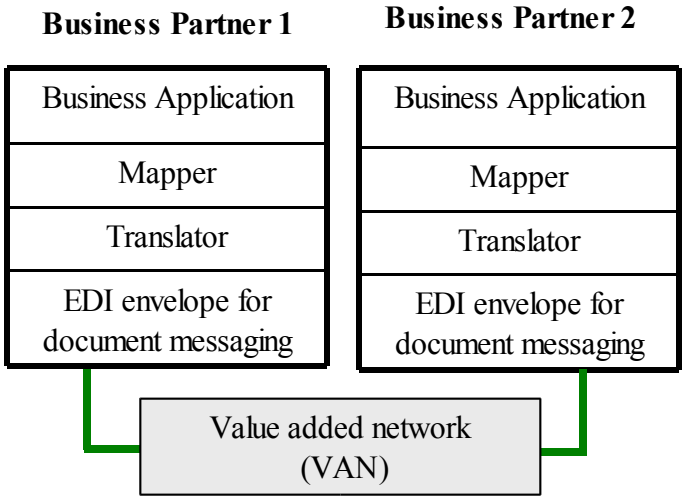
2. BizTalk



3. cXML



4. RosettaNet



5. EDI

EDI Standards: Overview

- Application-to-application transfer of business documents (e.g., purchase orders, invoices, shipping notices, billing and payment information, etc.)
- Aim is to minimize the cost, effort, and time incurred by paper-based business transactions
- EDI documents are structured according to a standard (e.g., ANSI X12 and UN/EDIFACT)
- Trading partners exchange business documents via a Value-Added Network (VAN)
- EDI technology infrastructure: mapper software, EDI translator, communication software, VAN.

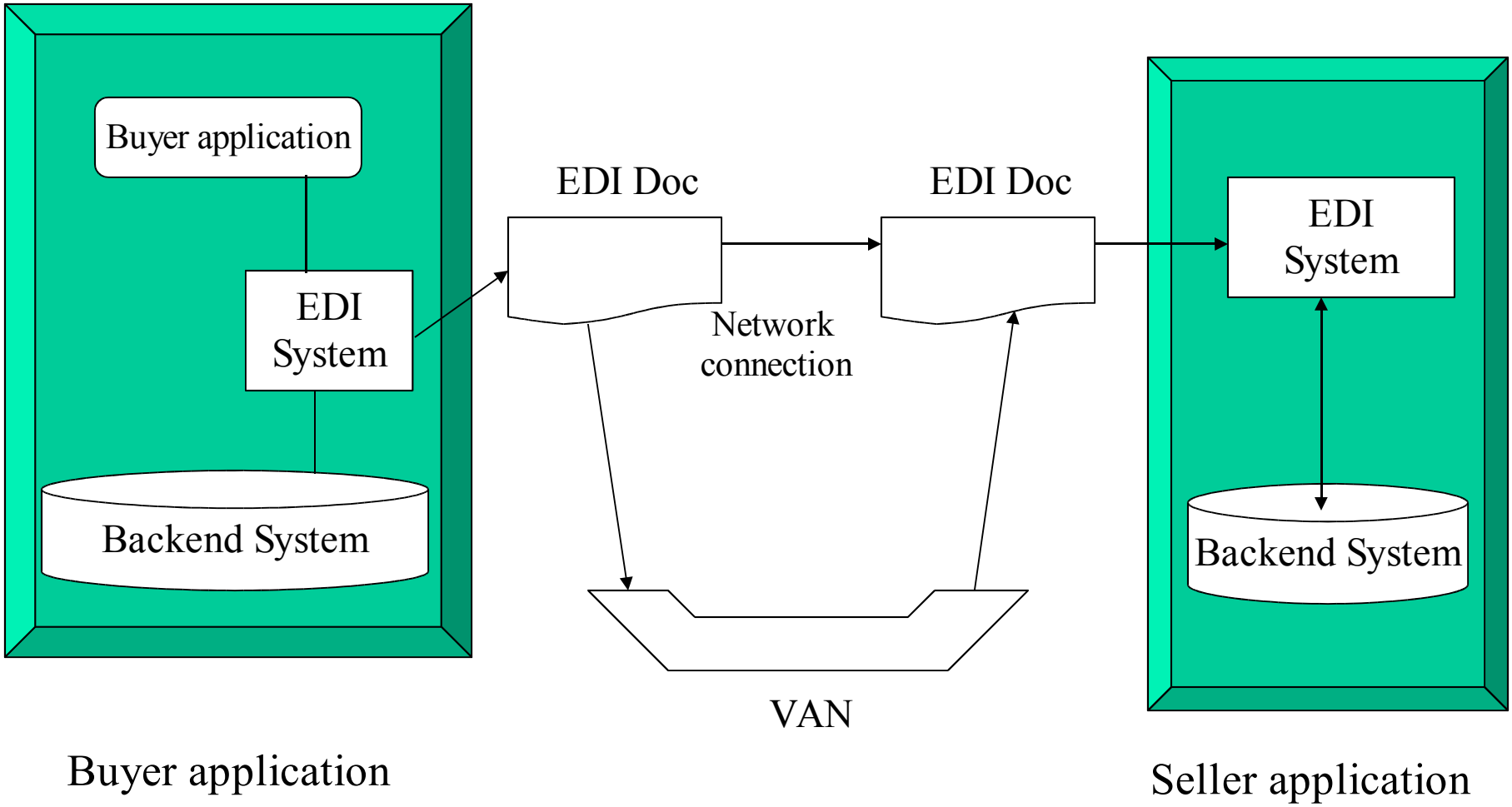
EDI messages

- Transaction Set (message): represents a business document (e.g., a purchase order), a set of segments
- Segment: a logical group of data elements (e.g., quantity, part description)
- Data element: basic fields such as order number, date
- Example: ASC X12 set number 810 is a transaction set for invoice (has 50 segments), 840 (quote for quotation), 855 (deliver order), etc.
- ANSI X.12
  - requires each element to have a specific name (e.g., invoice date, order date)
- EDIFACT
  - Terminology: message instead of transaction set in ANSI X.12
  - allows generic element (e.g., date)
  - fewer data elements and segments

EDI Solution components

- Business applications: generate and consume EDI messages
- Translation software
  - relationships between data elements in applications and EDI standards (e.g., transformation of a company-specific purchase order into EDI purchase order)
  - translators can be provided by third-party vendors or custom translators (in house)
- Communication software
  - manages and maintain phone numbers of partners, automatic dialling, up/downloading
  - Message envelope: contains a destination address, transaction type
- Value Added Networks (VANs): communication (mailboxes), access control, document tracking, message routing

Architecture



EDI Standards: discussion

- Several benefits including cost and time saving in document handling

- Major limitations:
  - The cost of implementing an EDI solution is high: expensive and proprietary networks, ad-hoc development
  - EDI standards are not flexible: e.g., the introduction of a new type of a business transaction is complex and time consuming
  - Translation from/to EDI messages: standards are very complex to implement, industry group implementation guidelines

EDI: Interoperability layers

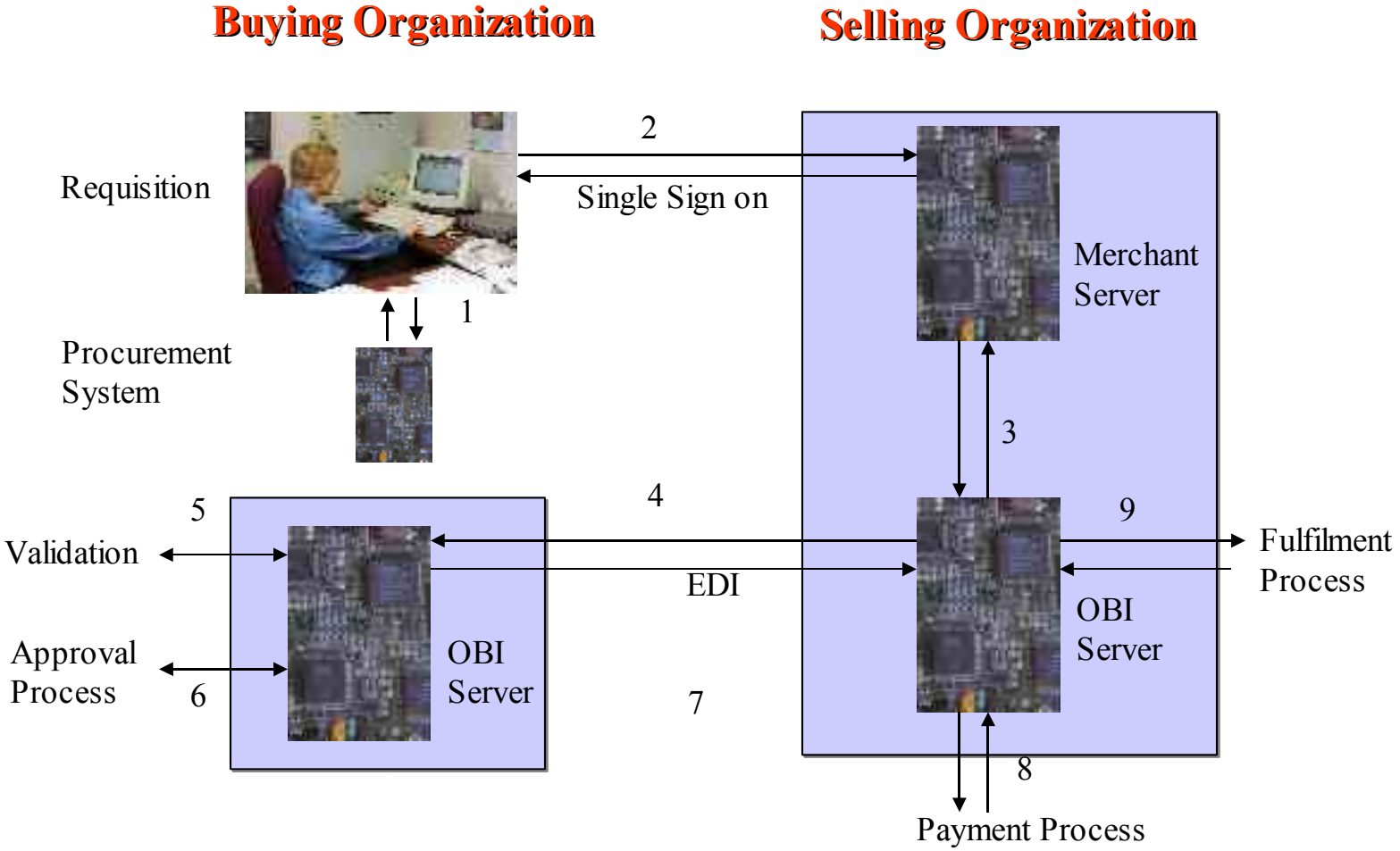
- Focus on communication and content interoperability
- VANs are used to handle message delivery and routing
- EDI standards provide a single homogeneous solution for content interoperability: the set of supported document types is limited
- EDI is very limited to enable a rich set of possible B2B interactions
- EDI standards, as currently defined, do not support interoperability at the business process level

EDI: B2Bi dimensions

- *Security:* private networks, document exchange (no need to reference partner systems)
- *Heterogeneity:* all partners are required to comply to the EDI standard
- *Scalability*: The cost of establishing a new relationship may be significant (partners may need to agree on the implementation conventions, integration of an EDI system with partner applications)
- *Adaptability*: EDI is inherently inflexible in its ability to adapt to changes (e.g., introduction of a new document type is complex and time consuming), impact of local changes is limited as partners do not directly reference each other systems

Open Buying on the Internet (OBI)

- Leverages EDI to define an Internet-based procurement framework
- Targets only non-strategic transactions: maintenance, repair, and operations (MRO) materials, office supplies, laboratory supplies, etc.
- OBI relies on the ANSI X12 EDI standard
- OBI objects: EDI messages (order requests/orders) and non-EDI messages (e.g., digital signatures of buyers and sellers)



Open Buying on the Internet (OBI)

OBI: Interoperability layers

- The HTTP protocol is used as a communication protocol
- Content layer
  - OBI objects
  - Buying organization: information about requisitioner profiles, trading partners, tax status, and approvals
  - Selling organization: dynamic catalogue (product and price)
  - OBI does not introduce a specific standard for describing buyer and seller catalogues.
- OBI defines a simple and pre-defined operational model for Internet-based purchasing
  - Commonly agreed upon activities (e.g., select a supplier, create order, send order request, receive order request, complete order, send order, process order, etc.)
  - Specifies only the way OBI servers communicate

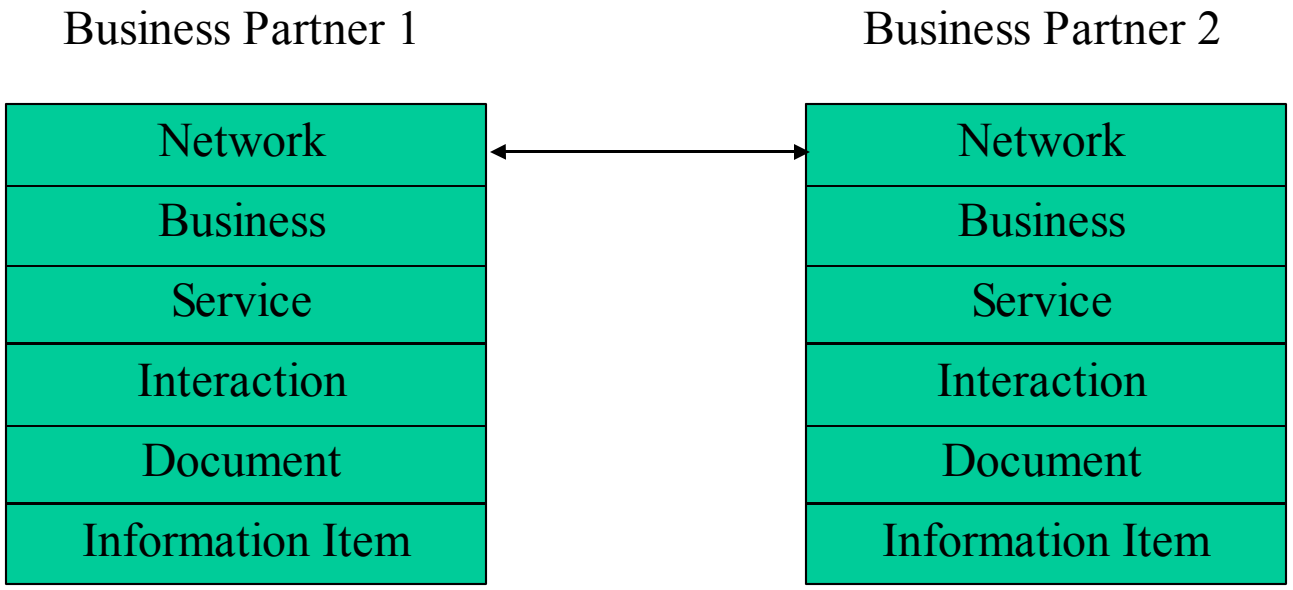
OBI: B2Bi dimensions

- *Security*: document exchanges, Secure Sockets Layer (SSL) over HTTP, digital signatures, digital certificates

- *Scalability and adaptability*: rates higher than EDI
  - Extensibility of order documents is not an important problem (OBI targets simple purchasing transactions)
  - OBI offers a lower entry cost (an Internet-based framework)

- Offers the same kind of properties as EDI with regard to the other dimensions.

eCO



- Aim: means for businesses to discover each other and access each other services regardless of the EC standards and protocols

- A layered architecture: discover partners, understand their offers, collaborate with them

eCO: Interoperability layers

- eCO uses XML-based schemas (currently, the Common Business Library - xCBL) to specify business documents (data and interfaces of services)
- At the process level, eCO focuses more on providing a common basis for business interactions. It does not focus on global business process definition
- Businesses advertise their services as Business Interfaces Definitions (BIDs)
- BIDs specify business services in terms of documents they accept and produce

eCO: B2Bi dimensions

- *Heterogeneity*
  - structural heterogeneity: uses XML schemas
  - semantic heterogeneity: very complex because EC industry covers broad area. Meta-data about layers help. The use of vertical (e.g., RosettaNet) and horizontal (e.g., OBI, EDI) ontologies will also help (but needs data normalisation, mapping and conversion between schemas)

- *Autonomy*: separation of the description of services and their implementations, common convention in marketplaces may impact negatively partner autonomy

- *Security*: document exchanges, use of security mechanisms is optional

eCO: B2Bi dimensions (Cont.)

- Scalability
  - A new relationship with a partner A: does not requires additional work for A
  - A new service: description of document types and service interfaces, integration of interfaces with internal applications
  - Cost of integration tend to be less significant in XML-based approaches: XML is a simple language, available XML processing and integration tools

- Adaptability
  - Impact of local changes is limited: partner systems are loosely coupled
  - eCO offers extensibility to accommodate changes: new descriptions can constructed by reusing and adapting existing ones

cXML

- Targets non-strategic transactions: maintenance, repair, and operations (MRO) materials, office supplies, laboratory supplies, etc.
- A simplified, XML and Internet-based version of EDI
- Assumes the existence of trusted third parties hubs (e.g., Ariba Network) between procurement and supplier systems
- cXML does not prescribe a specific third party architecture

cXML: Interoperability layers

- Two communication models
  - Request-response: synchronous, over HTTP
  - One-way: asynchronous, over HTTP

- Content layer:
  - cXML defines a set of XML DTDs to describe order documents (e.g., order request, order response)
  - Product catalogues: elements **Supplier** (general information), **Index** (inventory), …

- Business process layer:
  - Similar to OBI
  - Trusted hubs provide means for catalogue and order management (e.g., catalogue publishing, order routing and tracking)

cXML: B2Bi dimensions

- Offers the same kind of properties as OBI with regard to heterogeneity, autonomy, and adaptability
- Appears to rate higher than OBI with regard to scalability: integration cost in an XML approach tend to be less significant
- Security: document exchanges, cXML message headers include authentication information

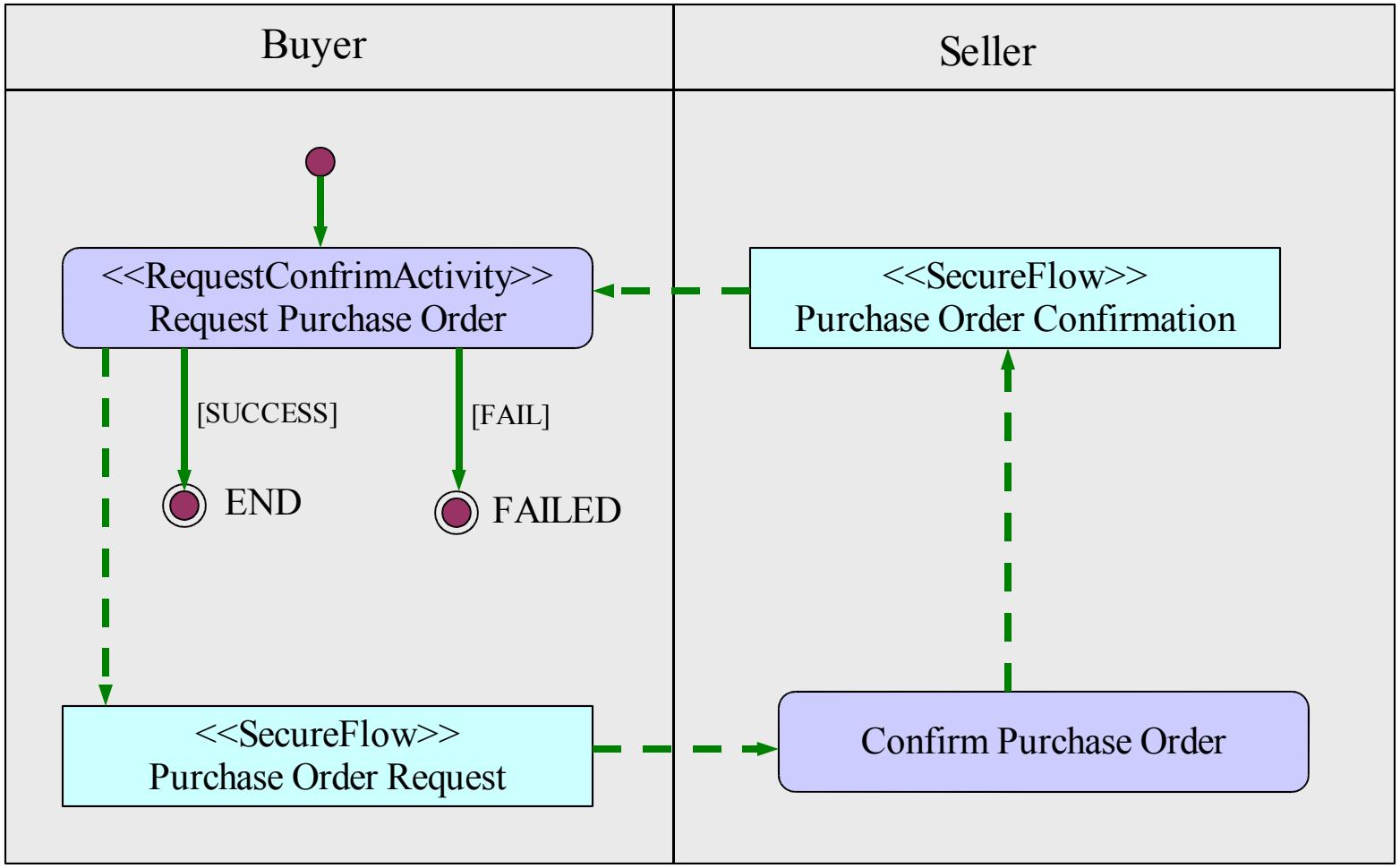
RosettaNet

- XML-based standard interfaces for supply chain management in information technology and electronic component industry
- Partner Interface Processes (PIPs) specify the processes and data elements by which partners can interact
- Technical and business dictionary: message's vocabulary, characteristics of products (e.g., computer parts), catalogues, business properties
- Implementation framework: message format, content, transport and security mechanisms

Some PIPs (Cluster 3 Order Management)

- **Segment 3A Quote and Order Entry**
  - PIP 3A1: Request Quote
  - PIP 3A3: Request Shopping Cart
  - PIP 3A4: Request Purchase Order
  - PIP 3A5: Query Order
  - PIP 3A6: Distribute Order
  - PIP 3A7: Notify of Purchase Order
  - PIP 3A8: Request Purchase Order Change
  - PIP 3A9: Request Purchase Order Cancellation
  - PIP 3A10: Notify of Quote Acknowledgment
  - PIP 3A11: Notify of Authorization to Build
  - PIP 3A12: Notify of Authorization to Ship
  - PIP 3A13: Notify of Purchase Order Information
  - PIP 3A14: Distribute Planned Order

PIP3A4


Buyer
Seller
<<RequestConfrimActivity>>
Request Purchase Order
[SUCCESS]
[FAIL]
END
FAILED
<<SecureFlow>>
Purchase Order Confirmation
<<SecureFlow>>
Purchase Order Request
Confirm Purchase Order

RosettaNet: Interoperability layers, B2Bi dimensions

- Interoperability layers:
  - Uses XML to describe documents
  - Focuses more on providing a common basis for business interactions, via PIPs. It does not focus on global business process definition

- B2Bi dimensions
  - Offers the same kind of properties as OBI with regard to security
  - Heterogeneity: XML to describe the structure, vertical ontologies to describe the semantics of documents and interactions
  - Offers similar properties as eCO with with regard to the other dimensions

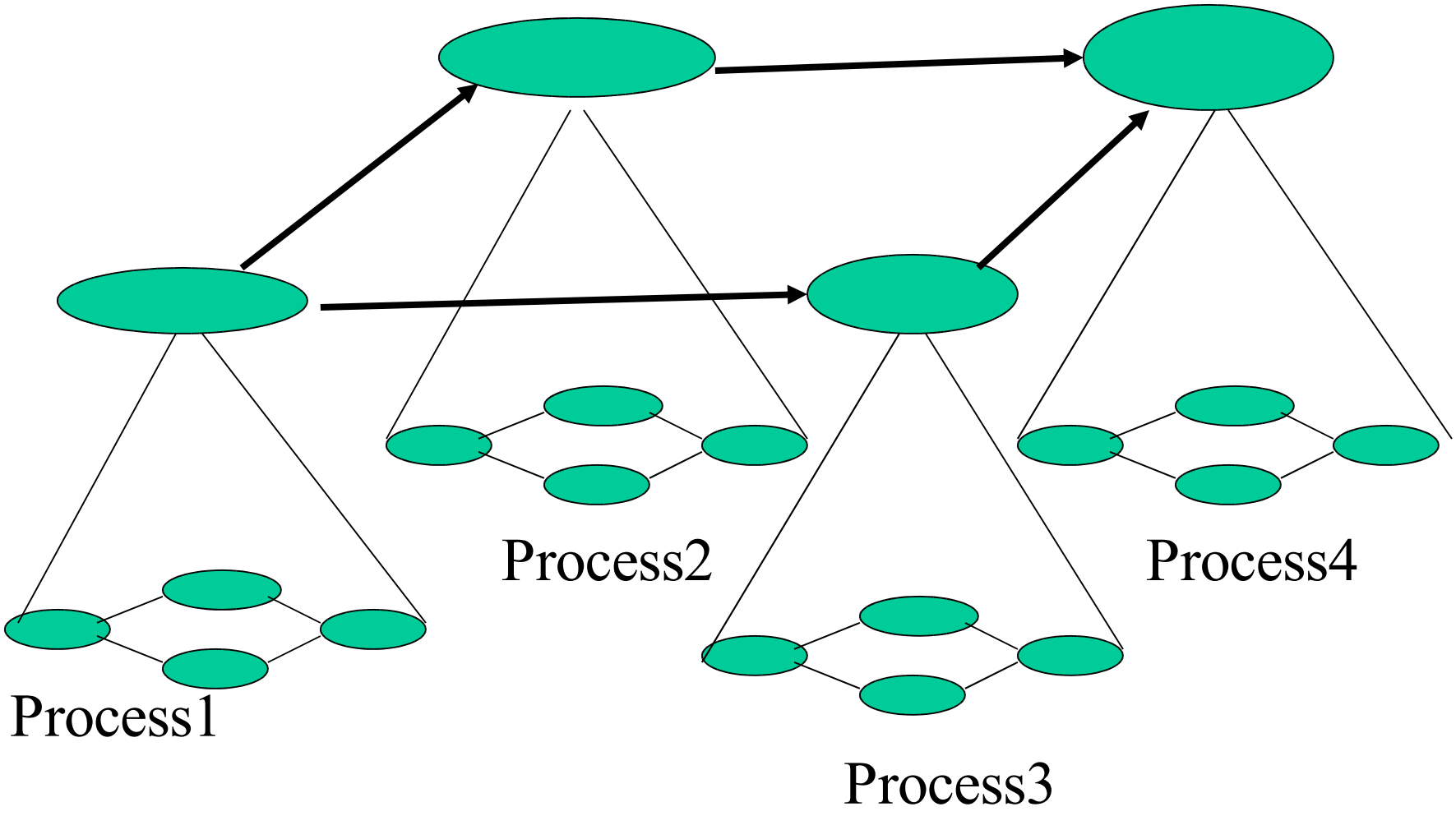
PART II: Integration Approaches

- Component-based Middleware
- Web Services
- B2B Integration Standards
- **Process-based Integration**

Process-based Integration : Overview

- Automation of business processes is an important enabler for applications integration both : within an enterprise and across partner systems
- Workflow technology is already a mature technology for automating intra-enterprise processes
- Traditional workflows: intra-enterprise, homogeneous and centrally controlled environments
- Inter-enterprise business processes:
  - support the collaboration among of diverse users, applications, and systems
  - automate business processes that integrate disparate applications and systems
  - different processes schemas, different execution engines

Inter-enterprise Business Processes



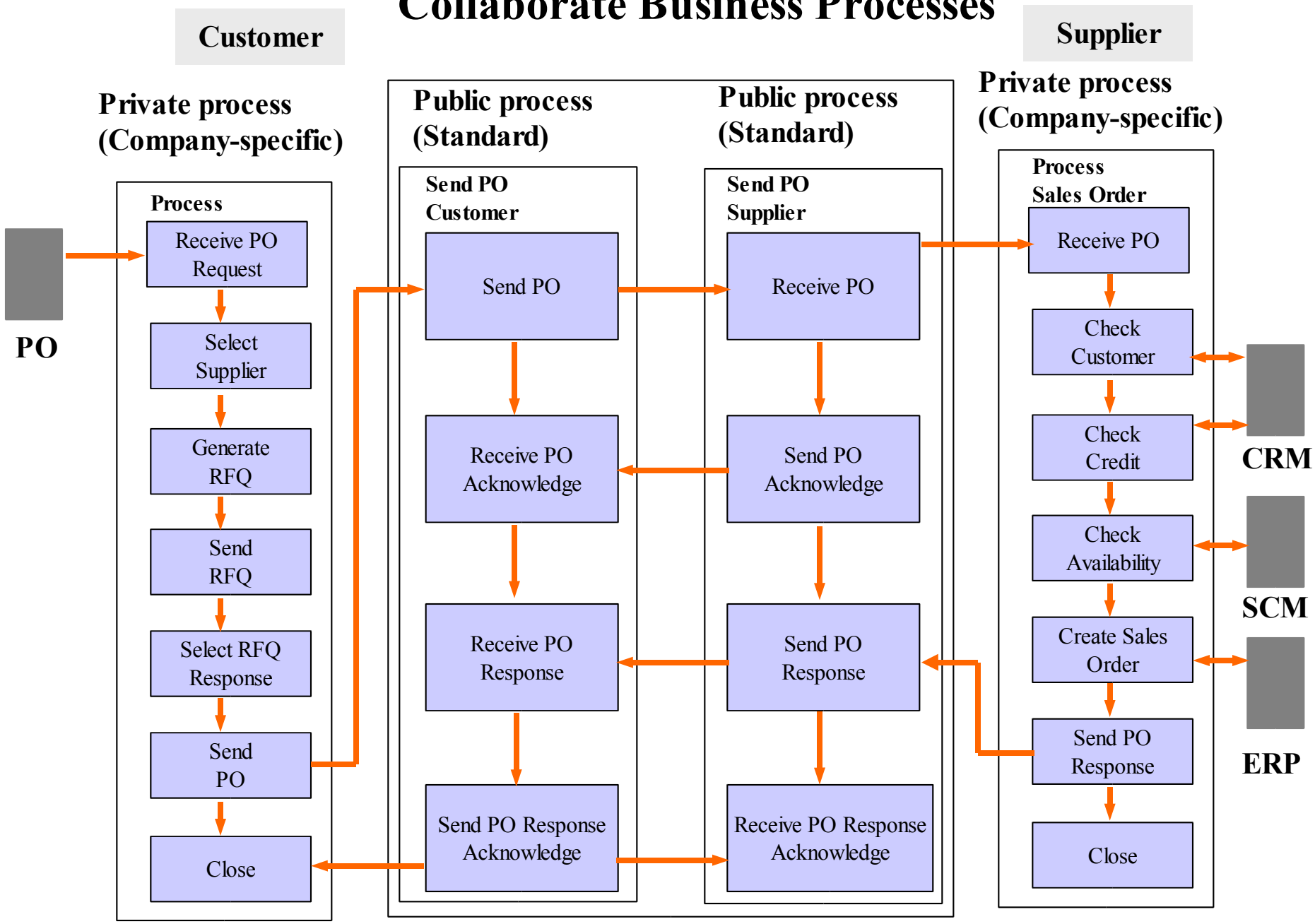
Inter-enterprise Business Processes: Is distributed workflows a solution ?

- Partitioning global workflows into sub-workflows
- Sub-workflow = activities that are to be executed by a unit (organization)
- Impose that each participant deploys a full-fledged execution engine capable of interpreting the workflow definition
- Same process model must be adopted by all participants
- Assume a tight coupling among sub-workflows
- Quite restrictive for B2B collaboration :
  - Partners may use disparate data and process representation models
  - Modifications of back-end applications, sub-workflows, and global workflow need to be coordinated.
  - The cost of establishing a new relationship may be significant (business processes must be modelled and deployed in concert)

Collaborative Process Management

- Separation between public and private processes
- Public process: external message exchange of an organization with its partners according to a message exchange protocol (e.g., EDI, RosettaNet)
- Private process: internal executable activities that support activities of public processes
- Private processes may also interact with lack end applications
- In this approach, there is no requirement that local execution engines be identical (e.g., one engine is based on IBM's MQSeries and another HP's Process manager)

Collaborate Business Processes



(Source: e-business Architectures and Standards, Anil L. Nori, Tutorial, VLDB'2002, HongKong, China)

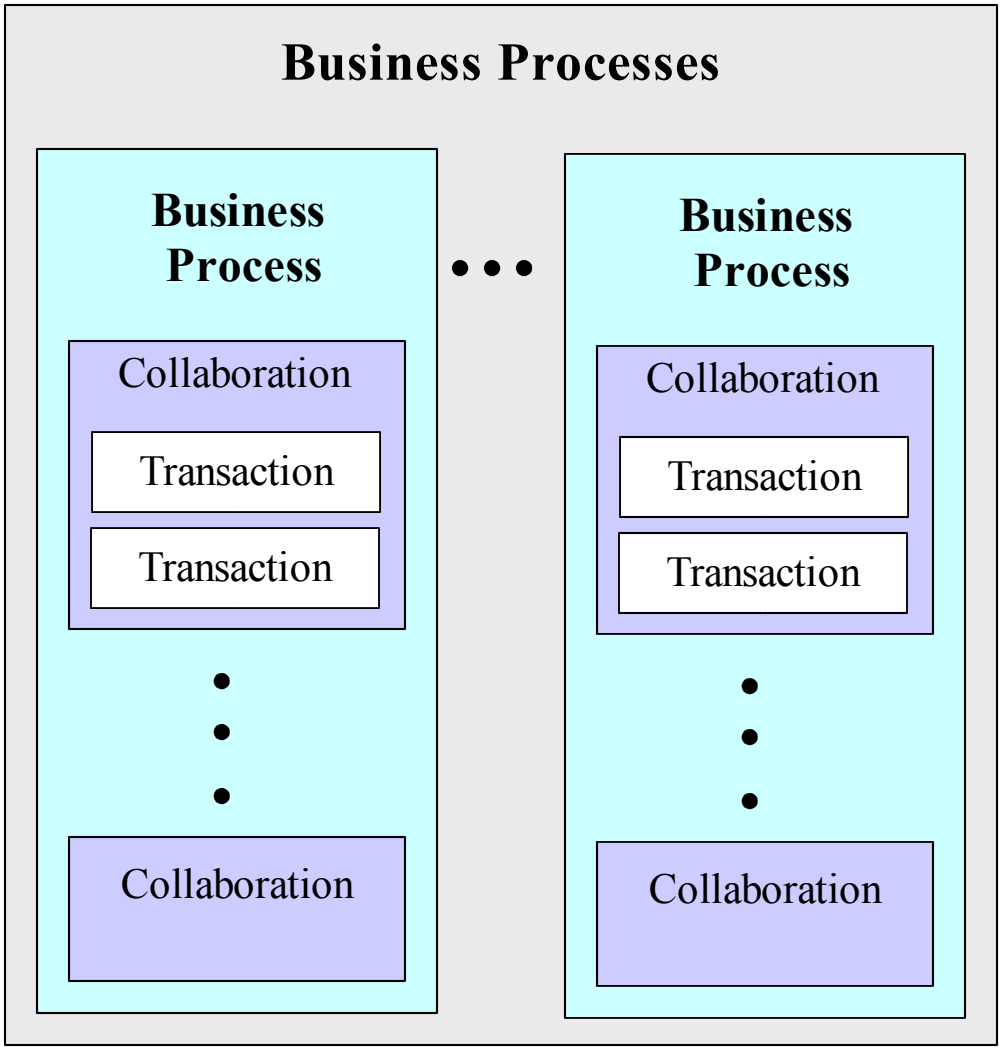
Collaborative Process Management (Cont.)

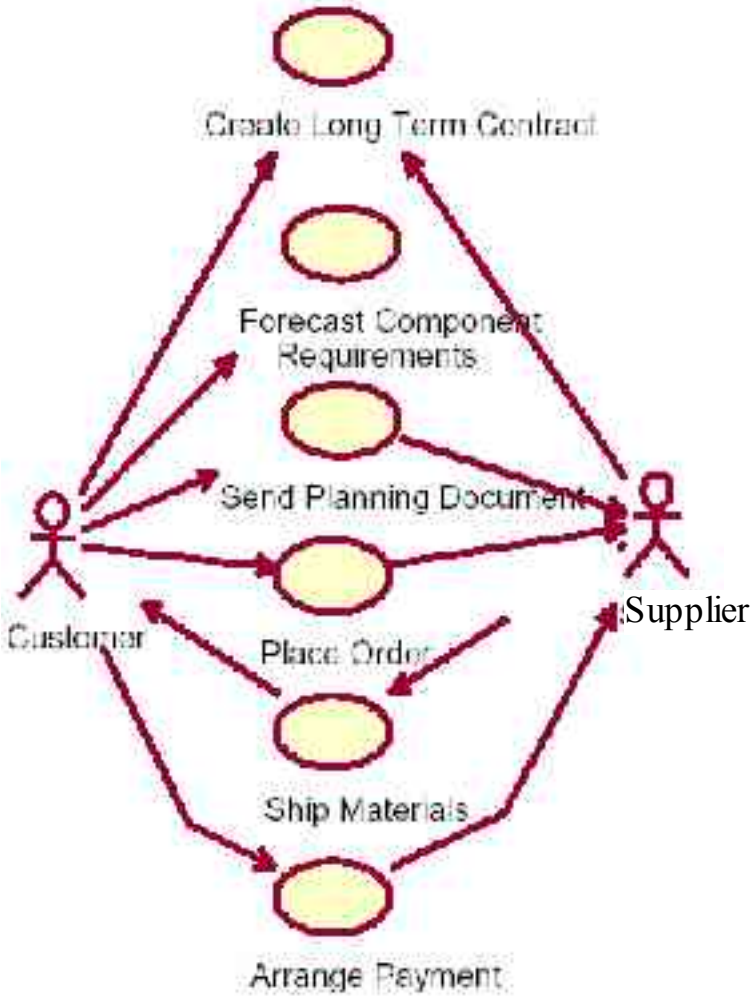
- Heterogeneity: semantic integration of processes is a difficult problem, incorporating process ontologies (e.g., RosettaNet) may help.
- Change propagation
  - Changes to private processes are local (separation of public/private)
  - Changes to interactions between local and global (e.g., formats of incoming or outgoing messages) may require modification of relationships between local and global processes
- Scalability:
  - The support of a new interaction protocol (e.g., EDI) requires the creation of a new public process and its relationships with the private process
  - The creation of a relationship with a new partner may require some adjustments (e.g., if the partner does not comply to an already supported interaction protocol, a new public process must be created)
- High level specification of business processes: fast integration

ebXML

- **ebXML BPSS** (**Business Process Specification Schema**)
  - Specifying collaborations
  - Collaboration = set of choreographed transactions
  - Transaction (activity) has one requesting document and an optional responding document
  - Binary collaboration (establish roles of partners), e.g., a Buyer can start the business transaction a seller can respond to it
- **ebXML CPP** (***Collaboration Protocol Profile***)
  - IT capabilities of a partner
  - Details of transport, security, messaging capabilities/constraints
  - References to supported business processes (ebXML BPSS documents) and roles that partner can play in these processes
- **ebXML CPA** (***Collaboration Protocol Agreement***)
  - Agreed upon capabilities in a collaboration
  - Can be generated from CPPs of partners (This may be involve negotiation between partners)

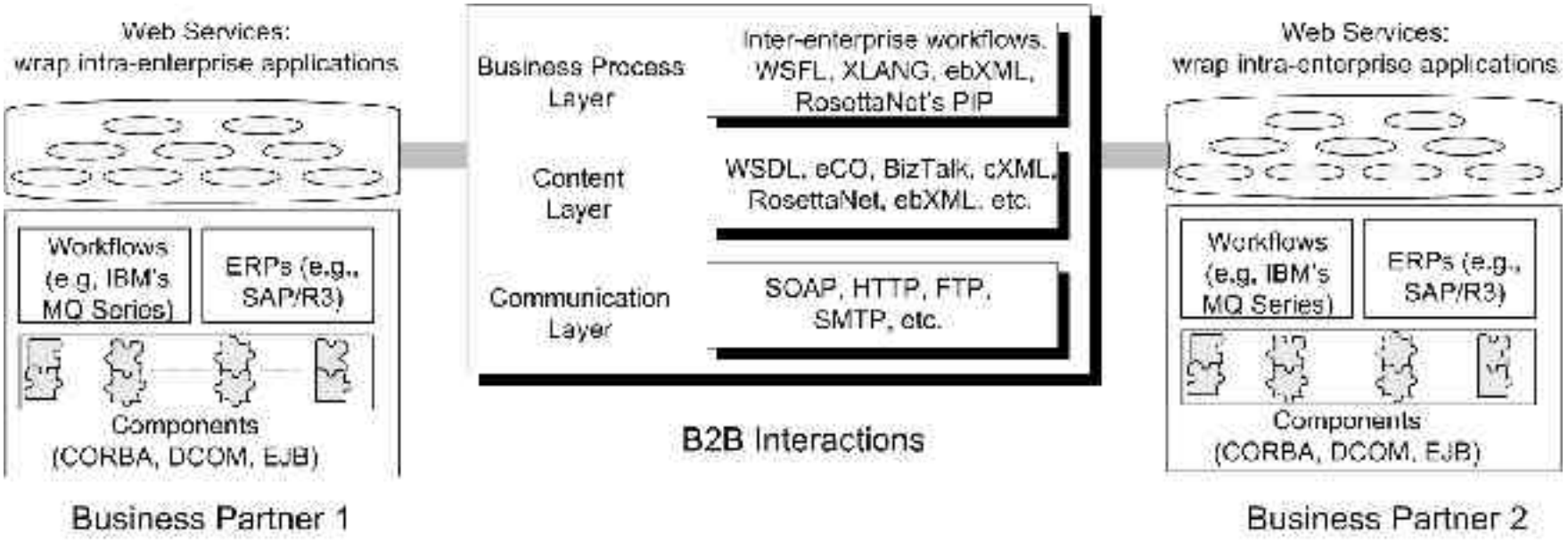
ebXML BPSS


Business Processes
Business Process
Collaboration
Transaction
Transaction
Collaboration
Business Process
Collaboration
Transaction
Transaction
Collaboration


Create Long Term Contract
Forecast Component Requirements
Send Planning Document
Customer
Supplier
Place Order
Ship Materials
Arrange Payment

Summary and Outlook

Putting Things Together



Summary

- Component middleware strength lies in the separation between applications and infrastructure services (e.g., persistence management, security management, transaction management, trading, event, naming services)
- Component middleware are suitable for building robust and secure applications within an enterprise *(tightly coupled integration - intra-enterprise integration, legacy applications)*
- Web services promise to take components step further by enabling loosely coupled inter-enterprise interactions (*XML/document-based)*
- Process-based integration is gaining considerable momentum. It provides an attractive alternative to hand-coding the interactions between applications using a general-purpose programming language
- B2B interaction standards such as EDI, RosettaNet can be used to define the semantics of business documents and interactions (*interactions semantics - standard vocabularies and business processes*)

Open Issues

- Convergence
- Dynamic and scalable orchestration of integrated services (number of services to be integrated may be large and continuously changing, decentralised coordination of service executions)
- Dependable and reliable execution of composite services (transaction support in highly autonomous environments is a difficult issue)
- Change management for composite services
- Security and privacy
- Web services conversations: semantics (transactional semantics, QoS attributes)