SOA
Service Oriented Architecture

Kim Horn
2007

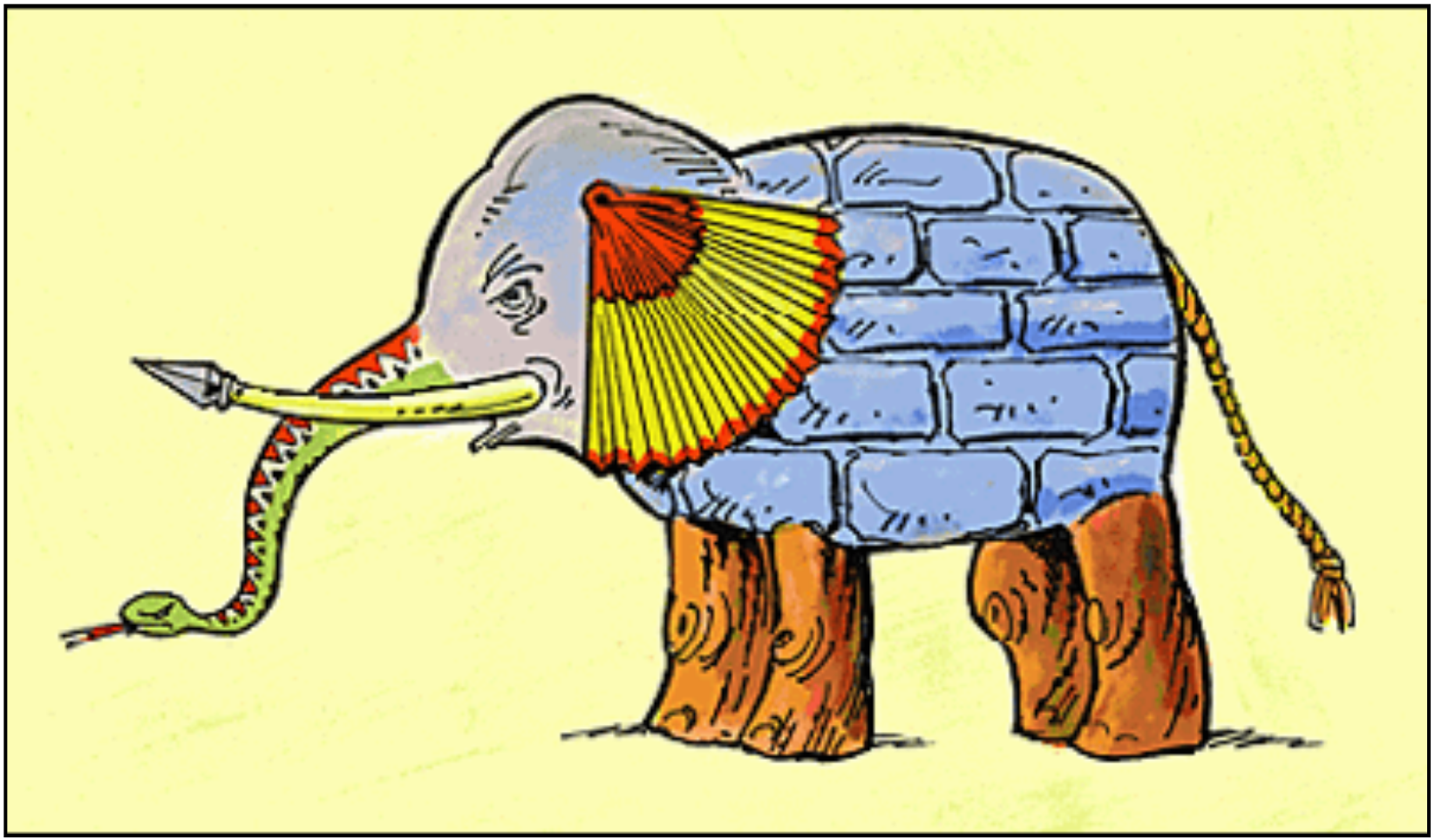
Agenda

"The single, most-important theme in modern application development is service-oriented architecture"

Gartner

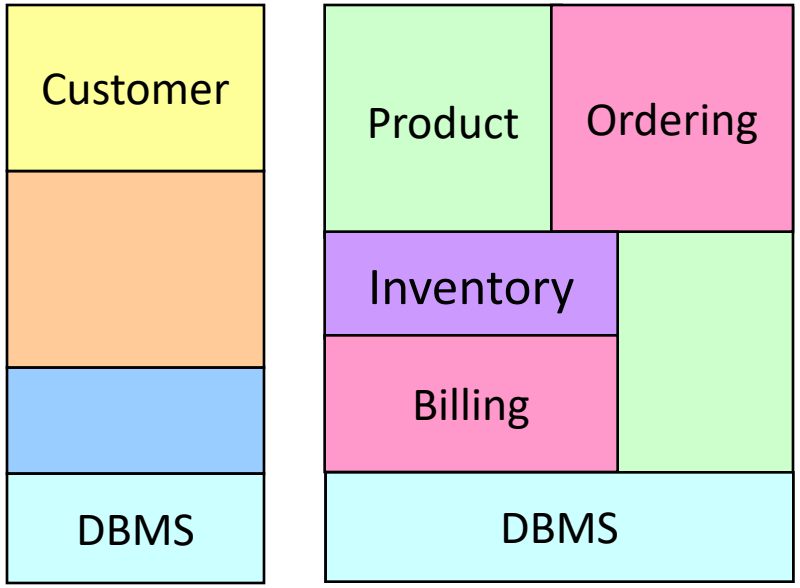
- **Brief Intro**: High Level Today
- **SOA Detailed**: Next Time
- **ESB Detailed**: Time After That

What is SOA ?

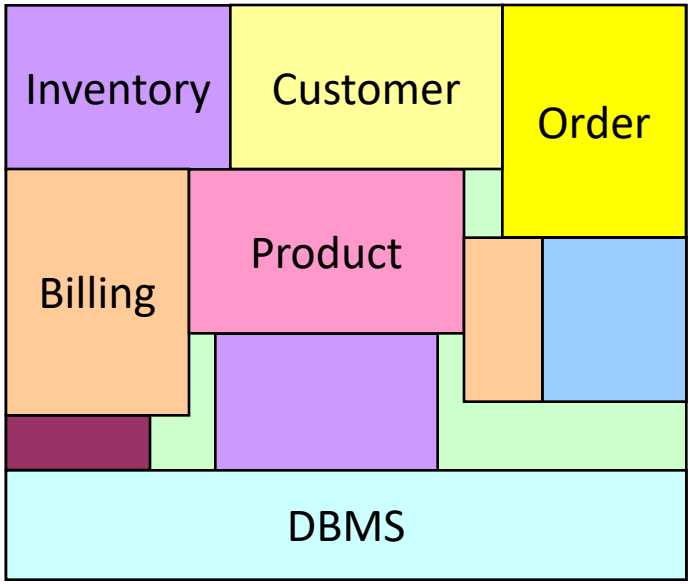


A Typical Organisations Systems

Division A

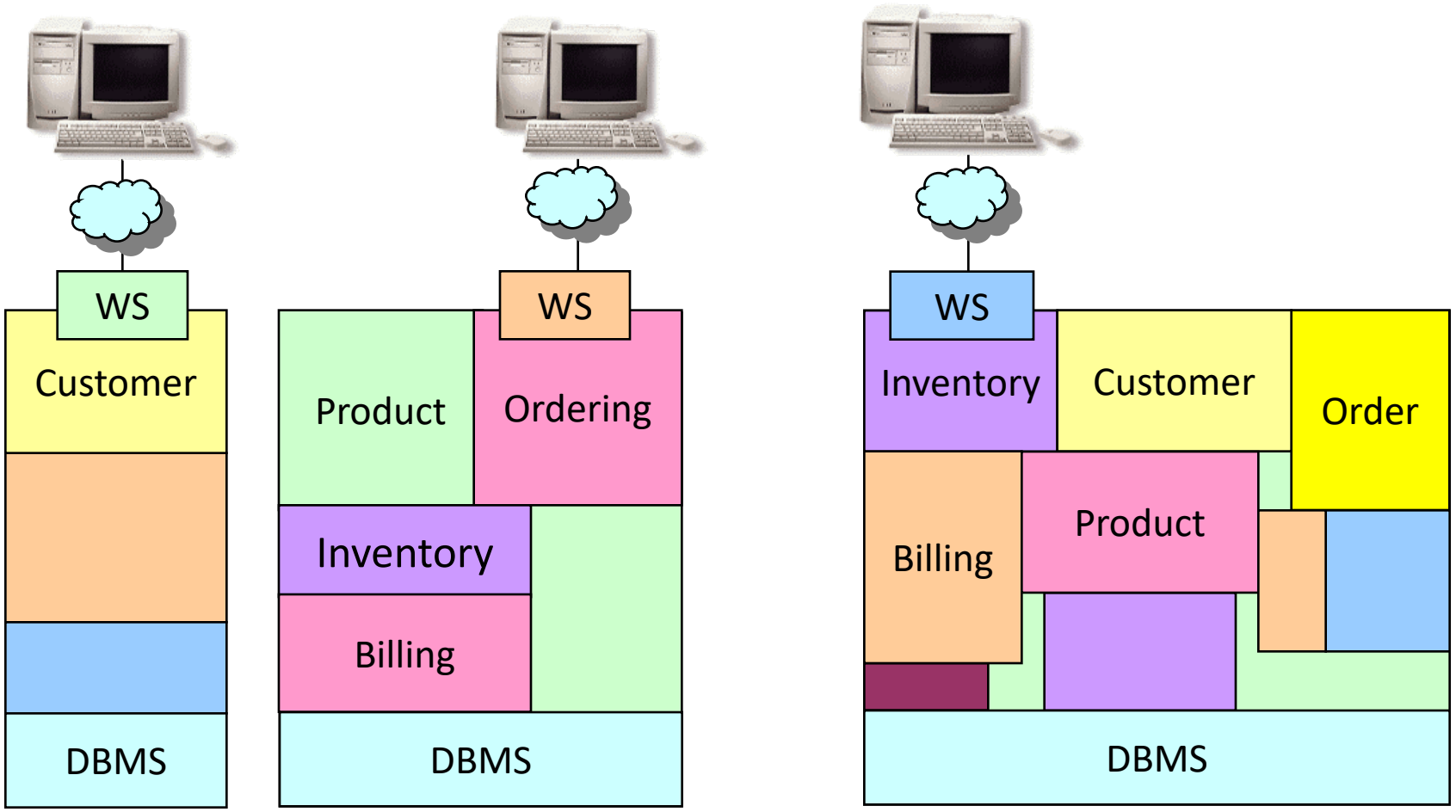


Division B



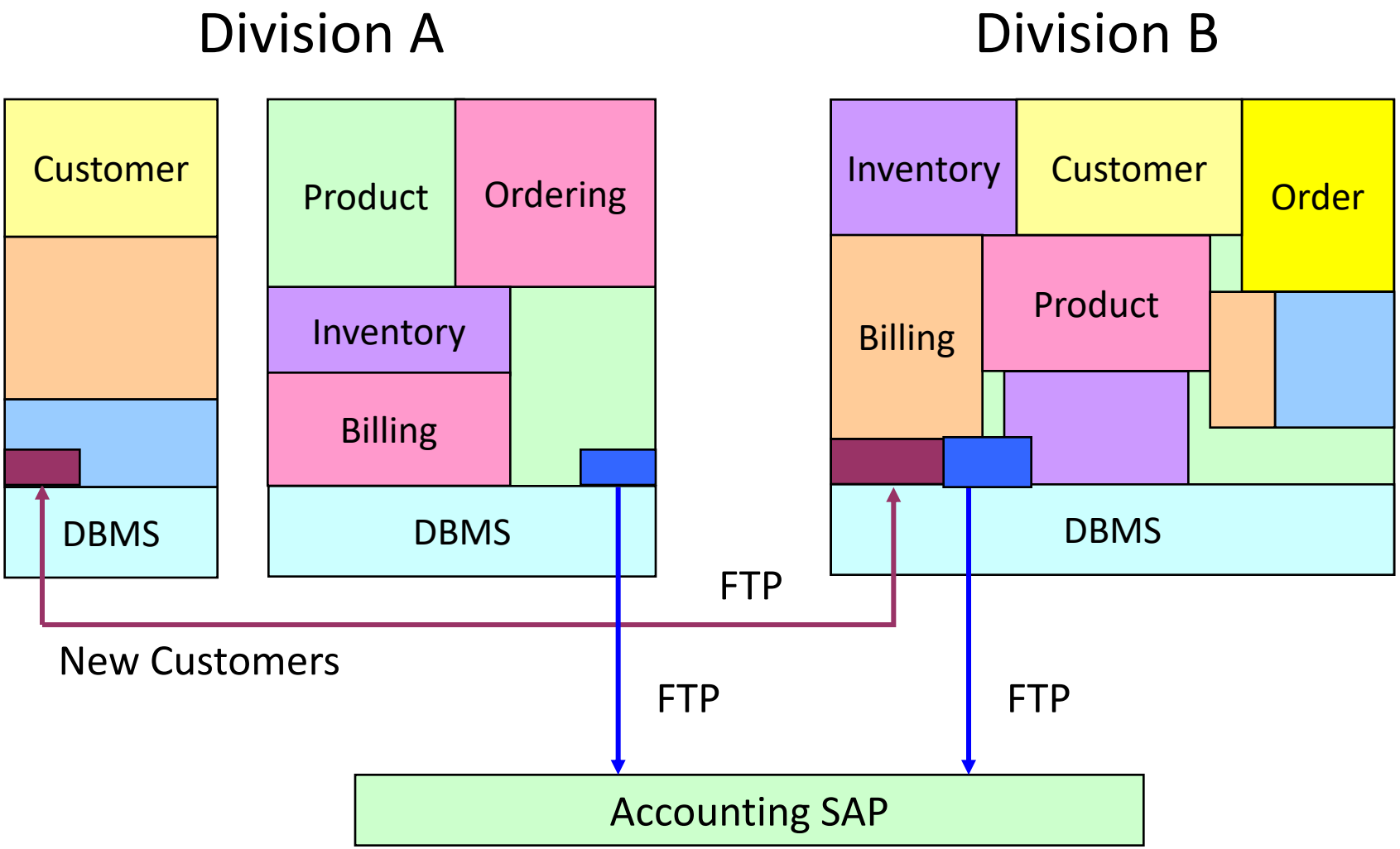
"Enterprise Fortress" and "Silos"

Open Up to The Web



"Web Silos"

Customer and SAP Integration



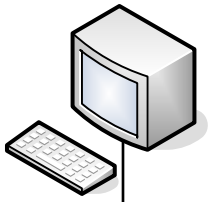
Issues with their Architecture

- Poor Integration, e.g. Customer data never gets properly aligned;
- Systems highly dependant and tightly coupled (can't change one without ramifications across many). Poor separation of concerns;
- Unreliable Services. Single points of failure. Difficult to Scale up system;
- Poor Reuse. Duplication instead;
- Many different platforms and out of date technologies used across systems;
- Slow and Expensive to build new applications and extend existing;
- Systems don't match the business processes;
- Difficult to Test;
- Maintenance is high;
- Limited ability to measure SLAs / Capacity Management;

Division A decides to rewrite systems.

WS

Customer App | Ordering App



Order Process App

Multiple Clients

2 Layers

Presentation

Business

API	API	API	API	API	API
Customer	Order	Product	Inventory	Billing	Utility
DBMS	DBMS	DBMS		DBMS	

Services API = Contract

Features of Service

Management Interface

Service Interface

Service

- Autonomous and Implementation independent;
- System and location independent;
- Based on well defined public interfaces (Open Standards);
- Can be shared and re-used;
- Capable of being assembled and reassembled to create new services;
- Coarse grained and few in number;
- Location and system independent;
- Provide predictable QoS;
- Combined, brokered, metered and outsourced
- Controlled and managed across the network;
- Idempotent

Services are Loosely Coupled: Messages

Tightly Coupled

Order App

change here

API Call, RPC: RMI, IIOP, Corba, Com, dCom……

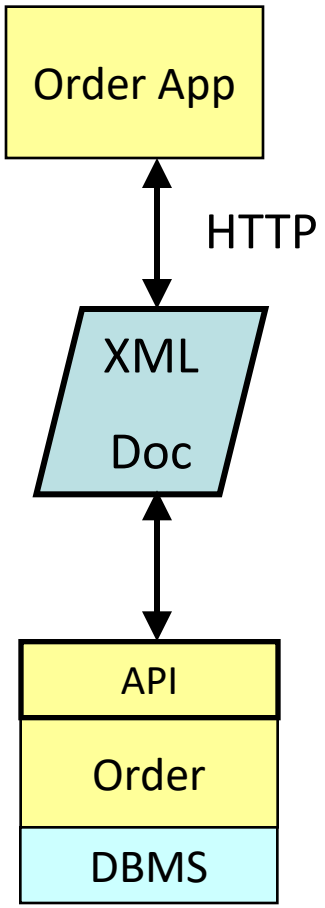
Order Code

DBMS

means change here & vica versa

Loosely Coupled

- Process is robust to changes in XML documents.
- Asynchronous
- SOAP Messages - Not Point to Point



Purchase Order Document

```
<PurchaseOrderDocument>
    <createDate>11-15-04</createDate>
<shipTo>
    <street>abc st</street>
    <city>abc city</city>
    <state>CA</state>
    <zipCode>99999</zipCode>
</shipTo>
<billTo>
    <street>abc st</street>
    <city>abc city</city>
    <state>CA</state>
    <zipCode>99999</zipCode>
</billTo>
<items>
    <itemname>Fan</itemname>
    <price>2</price>
    <quantity>200.0</quantity>
</items>
<items>
    <itemname>Blender </itemname>
    <price>1</price>
    <quantity>75.0</quantity>
</items>
</PurchaseOrderDocument>
```

Add Registry to Find Things

Division A

WS

Customer App

Ordering App

Order Process App

Registry

API

API

API

API

API

API

API

Customer

DBMS

API

Order

DBMS

API

API

Product

Inventory

DBMS

API

Billing

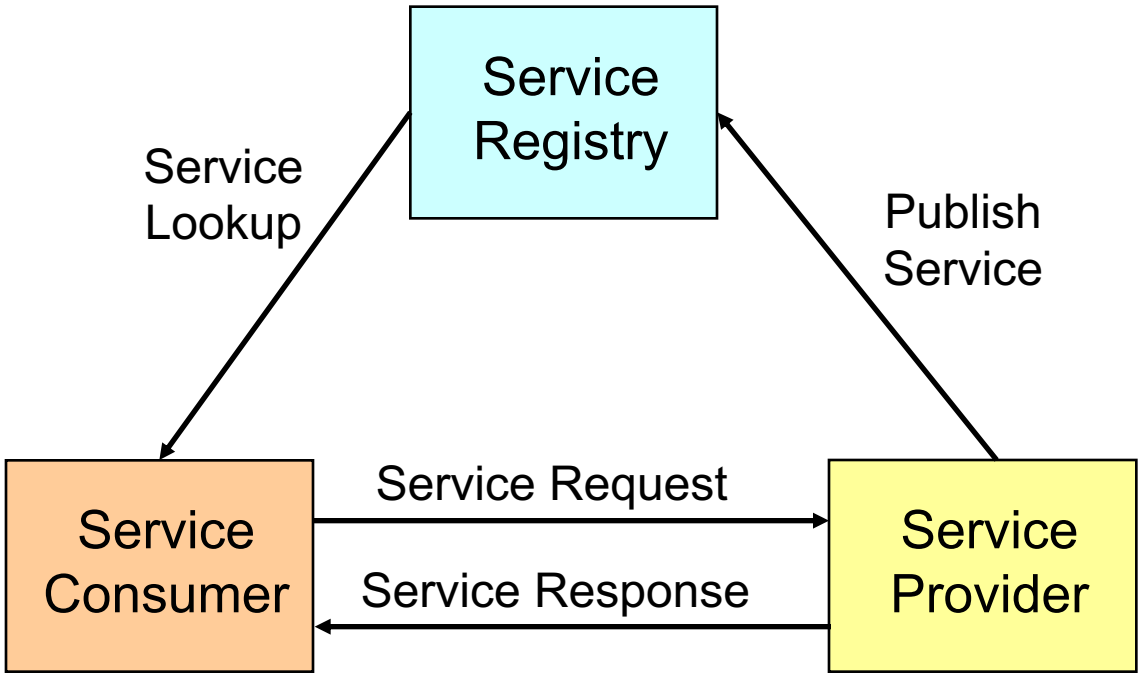
DBMS

API

Utility

FTP

An Early SOA Model*



SOA Operations

Who	What
Provider	Publish Service
Consumer	Find Service
Consumer	Bind Service
Consumer	Call Service

* New model described later…..

Web Service (Open) Standards

- **XML: Extensible Markup Language** – used to represent data (payload) and metadata in a standard way, used by all standards below;
- **SOAP: Simple Object Access Protocol** – provides the standard messaging format used by service consumers and providers. It is really just a thin wrapper around an XML application payload;
- **WSDL: Web Service Description Language** – used to describe a service, its location function and how to use it;
- **UDDI: Universal Description Discovery and Integration** – provides the standardised registry format, used to search for WSDL files.

The PO WSDL

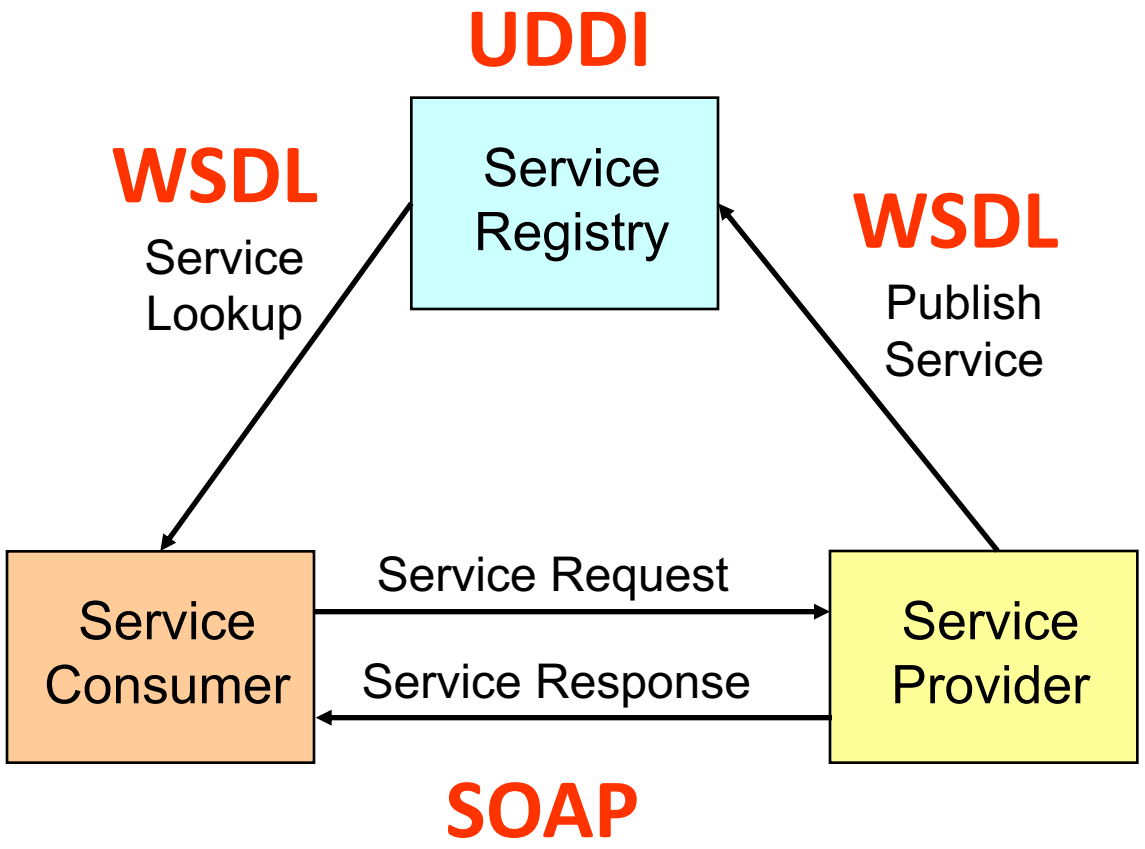
```
<schema targetNamespace="urn:SchemaDefinedPurchaseOrderService" ……..
    schemaLocation="PurchaseOrder.xsd"/>
   <element name="submitPO">
    <complexType>
     <sequence>
       <element name="inputDoc" type="pons:PurchaseOrder" nillable="true"/
     </sequence>
    </complexType>
   </element>
   <element name="submitPOResponse">
    <complexType>
     <sequence>
       <element name="result" type="string" nillable="true"/>
     </sequence>
    </complexType>
   </element>
   <element name="InvalidPOException">
    <complexType>
     <sequence>
       <element name="message" type="string" nillable="true"/>
     </sequence>
    </complexType>
   </element>
  </schema>
```

The PO Schema: PurchaseOrder.xsd

```
<xsd:complexType name="LineItem">
        <xsd:sequence>
           <xsd:element name="itemId" type="xsd:string" nillable="false"/>
           <xsd:element name="price" type="xsd:float" nillable="false"/>
           <xsd:element name="quantity" type="xsd:int" nillable="false"/>
        </xsd:sequence>
      </xsd:complexType>

<xsd:complexType name="PurchaseOrder">
    <xsd:sequence>
          <xsd:element name="poId" type="xsd:string" nillable="false"/>
          <xsd:element name="createDate" type="xsd:date" nillable="false"/>
          <xsd:element name="shipTo" type="Address" nillable="false"/>
          <xsd:element name="billTo" type="Address" nillable="false"/>
          <xsd:element name="items" type="LineItem" nillable="false"
    minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence/>
</xsd:complexType>
```

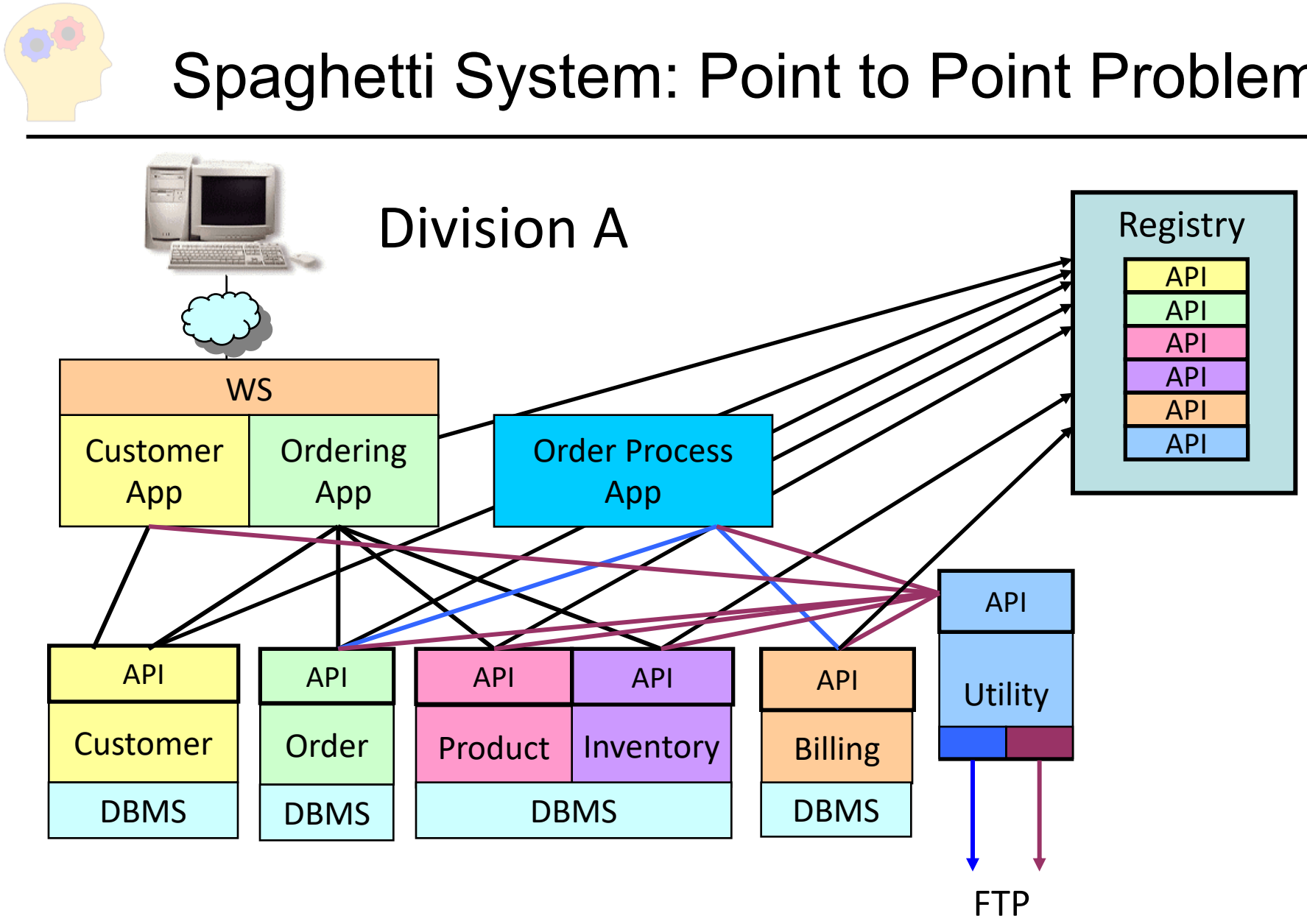
How Standards are Used



Registry can also contain:

- QOS
- Policy
- Security
- Metadata
- etc

Spaghetti System: Point to Point Problem



Solution: Introduce an ESB

Division A

Registry

API
API
API
API
API
API

WS

Customer App

Ordering App

Order Process App

ESB

API
Customer
DBMS

API
Order
DBMS

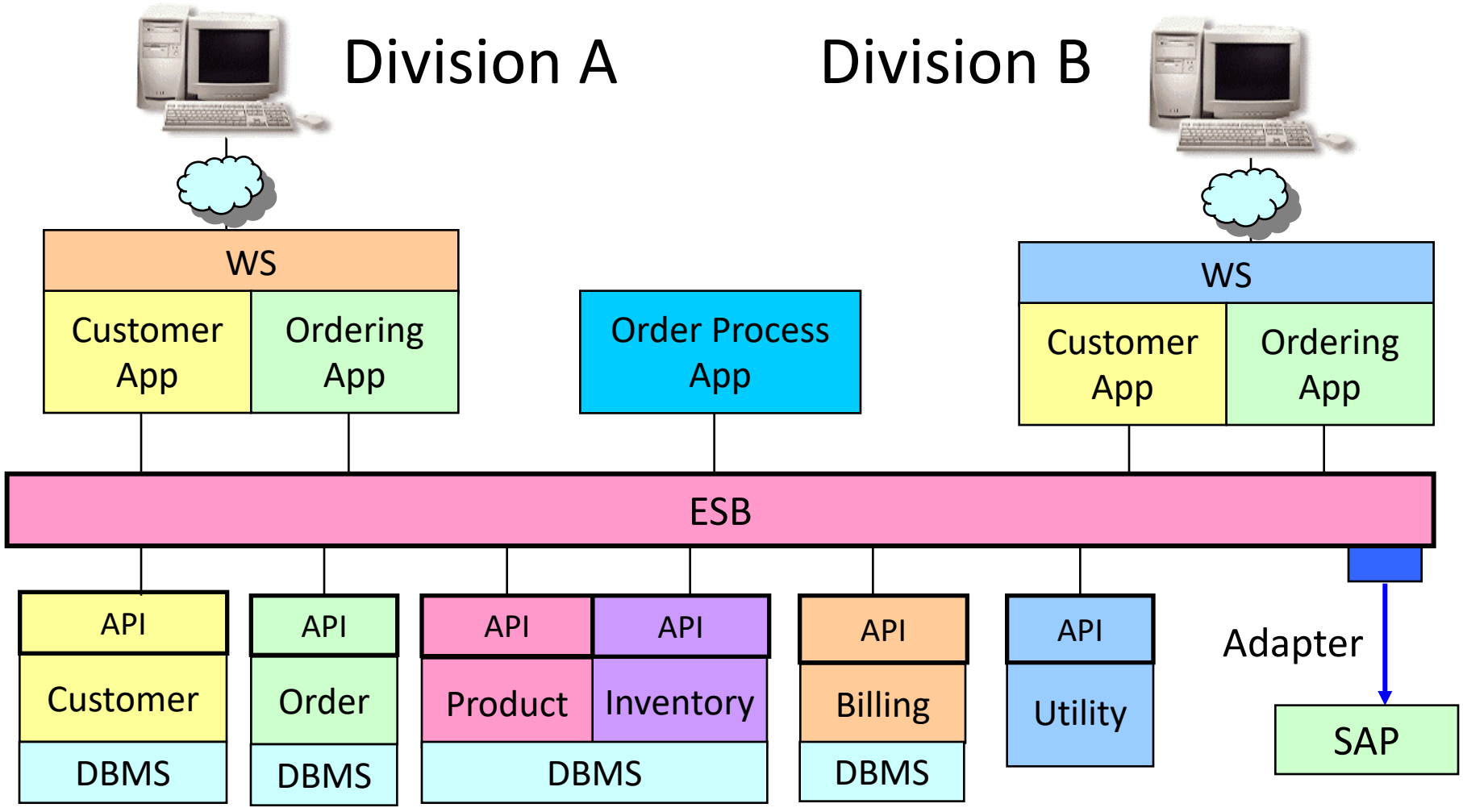
API
API
Product
Inventory
DBMS

API
Billing
DBMS

API
Utility

FTP

Now: Division B can Reuse A's Services



SOA Advantages

- Organisational agility and flexibility;
- Offer differentiated services and capabilities, quickly and easily;
- Deliver application architectures that mirror the business architecture;
- Leverage the legacy investment;
- Reduce costs by promoting reuse and better quality architectures
  - Applications are composed not built from ground up;
  - Loosely coupled, event driven;
- Improved and Open Integration:
- Across the internet;
  - Fix many of the problems with old EAI technologies.
  - Technical and philosophical disparities are blanketed.
  - Replace Portals to offer truly integrated solutions.
  - Use of open standards
  - Standardise on communication infrastructure.
  - Interoperability between platforms (e.g .Net and Java)

SOA Misconceptions

- SOA is new;
- SOA is OO;
- SOA is just a marketing term;
- SOA implies distributed systems;
- SOA is Web Services;
- A web service based application is Service oriented;
- Once you use SOA, everything becomes interoperable.

Answer is NO to all the above.

SOA Success Factors

- Recognise SOA is Architecture Style Not business panacea;
- Governance;
- Enterprise Architecture;
- Importance of Open Standards;
- Adopted in Stages;
- Well Defined Mature Process; and
- Requires Higher Availability.

Service Oriented Architecture

A Service Oriented Architecture (SOA) is a component model that decomposes a system into loosely coupled services that expose their functions through well-defined interfaces. The services are system and location independent and can be assembled and reassembled to create new services. Ideally services should provide predictable Quality of Service levels and be controllable and manageable across the network.

Kim Horn

A set of components which can be invoked, and whose interface descriptions can be published and discovered

W3C

A loosely-coupled architecture designed to meet the business needs of the organization.

Microsoft

SOA Definitions

Definition 1. SOA as seen by the developer and solution architect

A software-design approach in which key functions are built as reusable components that implement industry standards for interoperable communications.

Definition 2. SOA as seen by the enterprise architect

Enables loose coupling, interoperability, discoverability, management of change, and operation of business services in a well-governed environment. Business services operating in a well-run SOA can be composed into business processes that align IT with the business.

Keith Pijanowski, Platform Strategy Advisor
Microsoft Corporation

Definition of SOA – Open Group (Part 1)

- An **architectural style** that supports **service orientation**
- **Service orientation**
  A way of thinking in terms of services and service based development and the outcomes that services bring
- **Service**
  A logical representation of a repeatable business activity that has a specified outcome (e.g., check customer credit; provide weather data, consolidate drilling reports), is self-contained and may be composed of other Services. It is a black box to consumers of the Service
- **Architectural Style**
  The combination of distinctive features in which Enterprise Architecture is done, or expressed

Definition of SOA – Open Group (Part 2)

- The SOA Architectural style's **distinctive features**:
  - Based on the design of the services comprising an enterprise's (or inter-enterprise) business processes. Services mirror real-world business activity
  - Service representation utilizes business descriptions. Service representation requires providing its context (including business process, goal, rule, policy, service interface and service component) and service orchestration to implement service
  - Has unique requirements on infrastructure. Implementations are recommended to use open standards, realize interoperability and location transparency.
  - Implementations are environment specific, they are constrained or enabled by context and must be described within their context.
  - Requires strong governance of service representation and implementation
  - Requires a "Litmus Test", which determined a "good services"

Service Oriented Architecture

Service Oriented Architecture is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.

OASIS

SOA: A paradigm for organizing and utilizing distributed capabilities that may be under control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations.

Zapthink

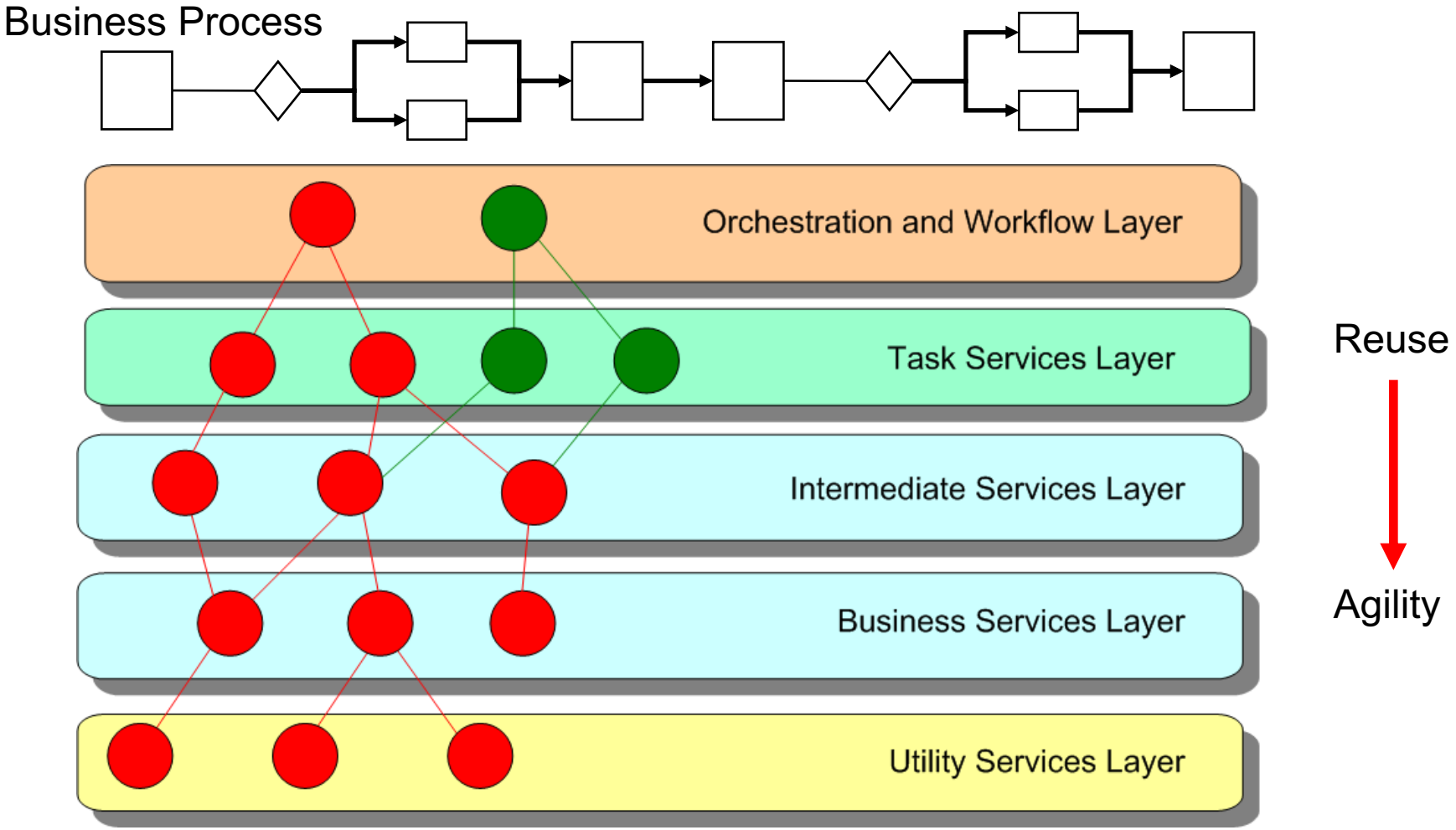
SOA is any architecture that can, at least on a logical level, be decomposed into three categories of components—a service, a provider of the service, and a consumer of the service.

Tarak Modi (www.ftponline.com/ea/magazine)

ESB Capabilities:

- Standards based **Integration** where application logic and integration logic are decoupled;
- **Messaging**
  - Across business boundaries, over the internet;
  - MOM and Web Services Standards;
  - Across Protocols: JMS, SOAP, JCA, WSIF, JDBC, EJB, RMI, HTTP, and FTP;
  - Configurable qualities of service and different types of persistence stores, throttling etc.
- **Routing**
  - Content based routing, and content filtering;
  - Business Process, Workflow and Itinerary based
- **Transformation/Validation**
  - Transformation, checking and document enrichment using XSLT, Xpath, XQuery, and/or Business Rules, etc
- **Adapters**
  - Provide standards based access to data sources and services.

Service Can Be: Layered & Composed

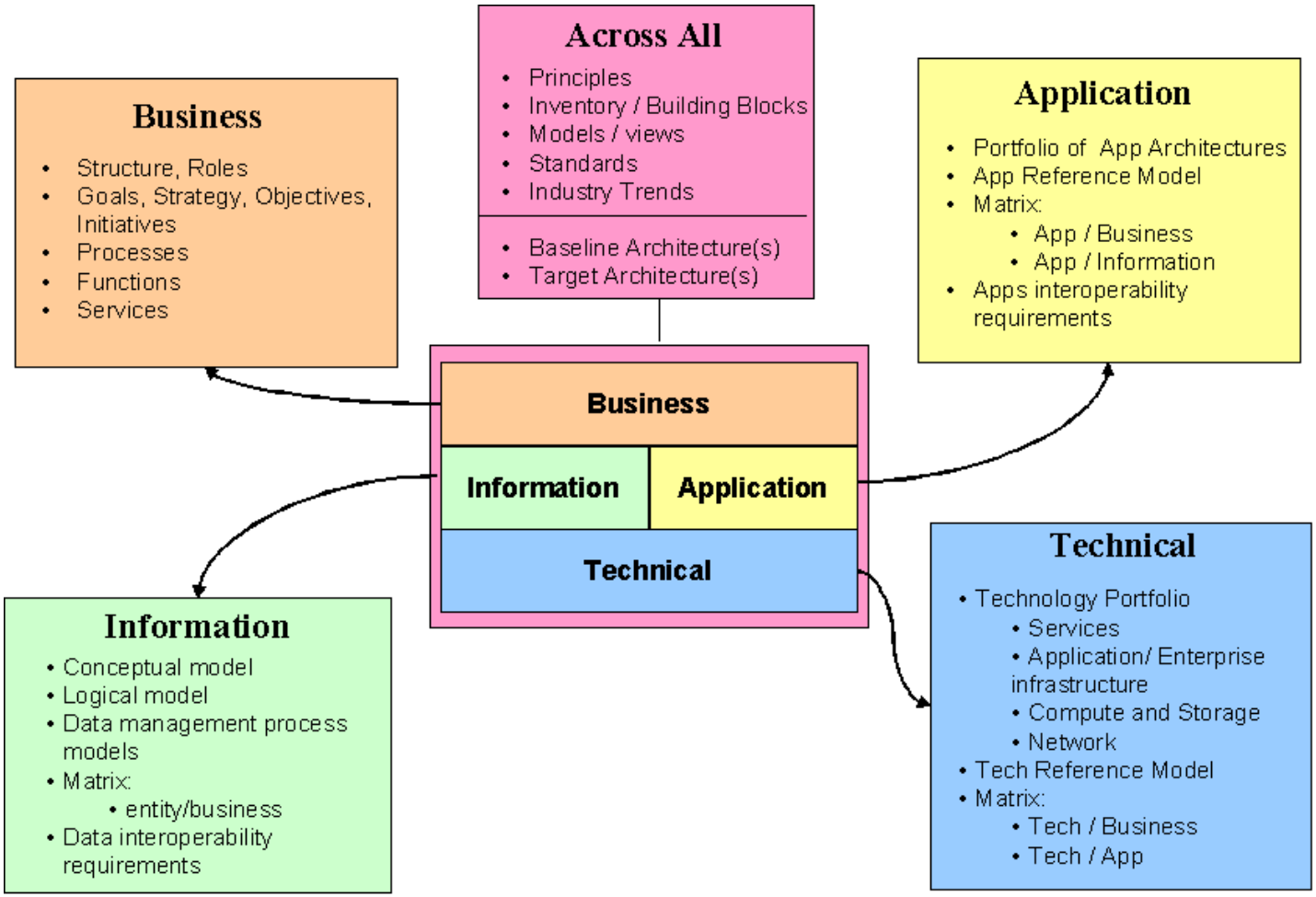


SOA Issues

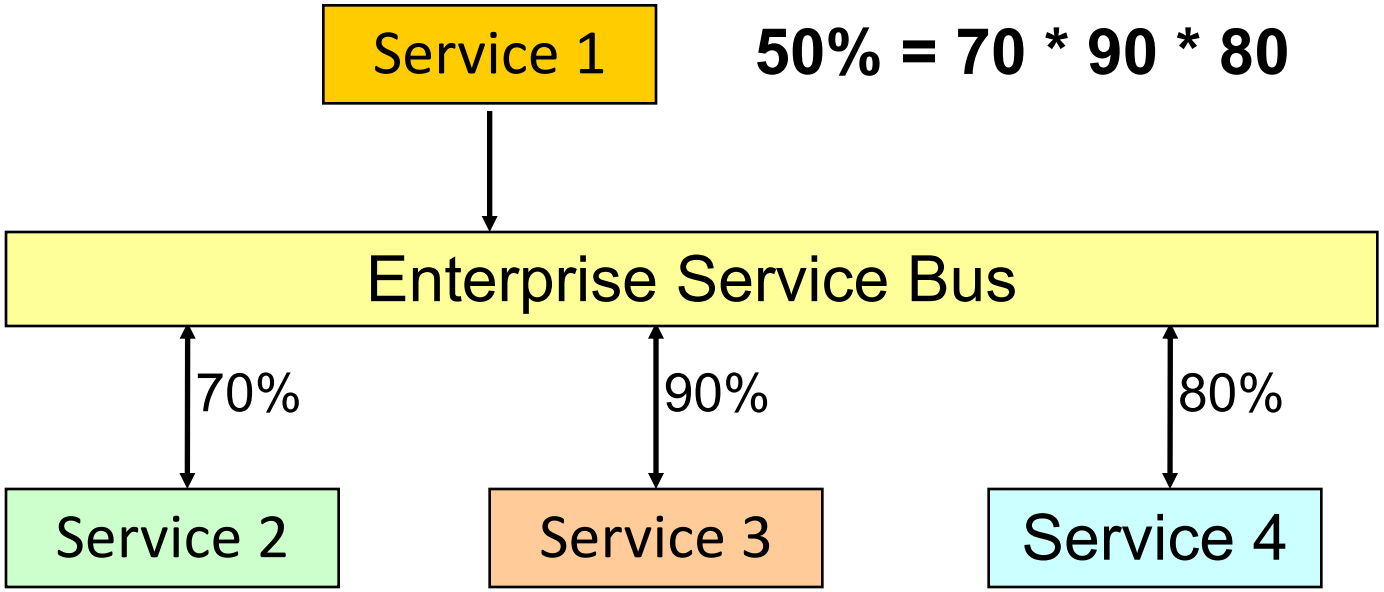
'Implementing an SOA is a stretch in skill set because you have to think about the business perspective first, understand which functions span across the organisation, and then design services around our business processes and our business architecture,'
Jeff Gleason, director of IT strategies, Transamerica Life Insurance Co

- Requires an Enterprise Wide approach that may extend to partners;
- Methodologies, both Enterprise and Solution, are immature;
- Services are hard to define (design);
- Technology/Standards in Security and Transactions still maturing;
- Standards are constantly on the move;
- Poorly architected applications make poor services;
- Transition planning is ignored; should be part of EA.
- Performance requirements are often ignored;
- Governance is often ignored;
- Hype.

Enterprise Architecture



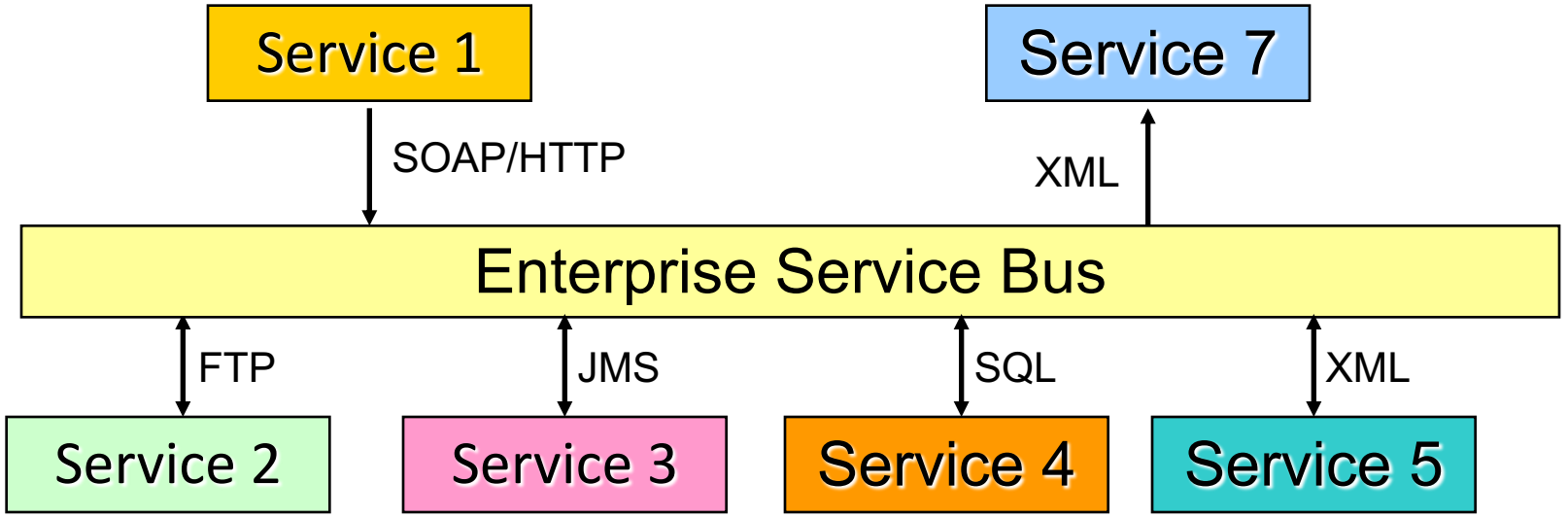
Availability



Service Communication

Enterprise Service Bus allows communication using Open Standards:

- Removes problems of point to point solutions;
- Document Driven, Loosely Coupled using XML, SOAP, Web Services;
- Asynchronous, Reliable, Event Driven, Mediation, Transformations, Adapters;

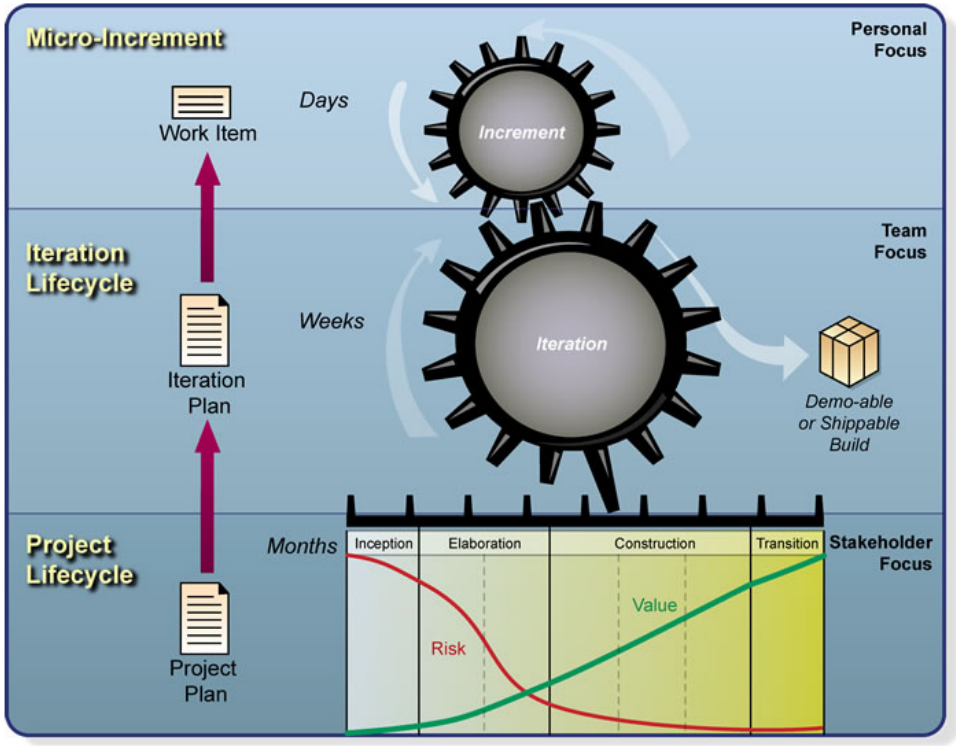


ESB supports 3 Integration Styles

1) **SOA** – applications communicate through re-usable services with well defined, explicit interfaces. Leverages two styles below:
2) **Message Driven Architectures** – applications send messages through ESB to consumers.
3) **Event-Driven Architectures** – applications generate and consume messages independent of each other.

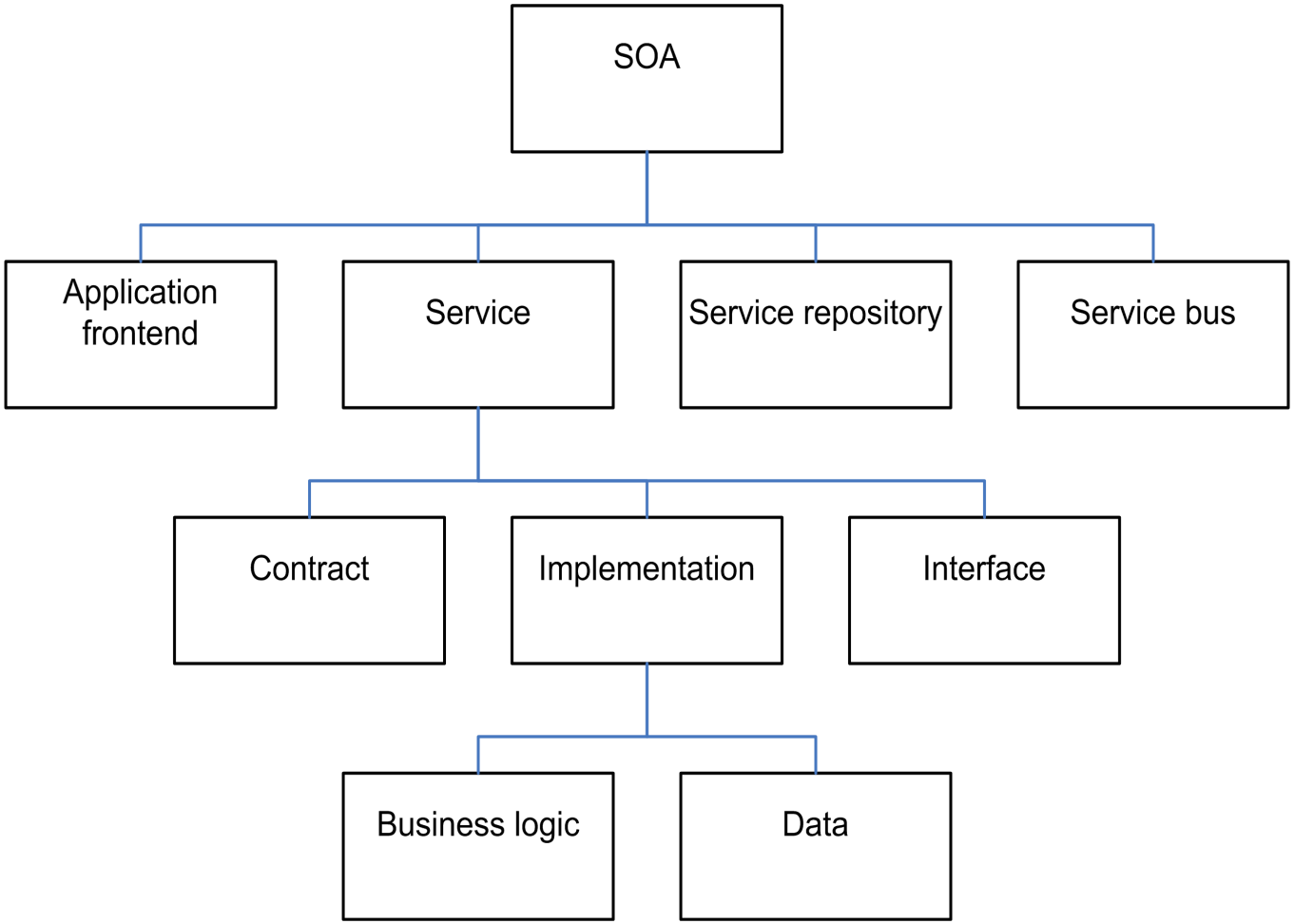
Process, Methodology, SDLC

- The Eclipse Process Framework (EPF) aims at producing a customizable software process engineering framework, with exemplary process content and tools, supporting a broad variety of project types and development styles.
- **Exemplary tool: EPF Composer**
- **Exemplary process: OpenUP**



OpenUP is a lean UP that applies iterative and incremental approaches within a structured lifecycle. OpenUP embraces a pragmatic, agile philosophy that focuses on the collaborative nature of software development. It is a tools-agnostic, low-ceremony process that can be extended to address a broad variety of project types.

Elements of SOA



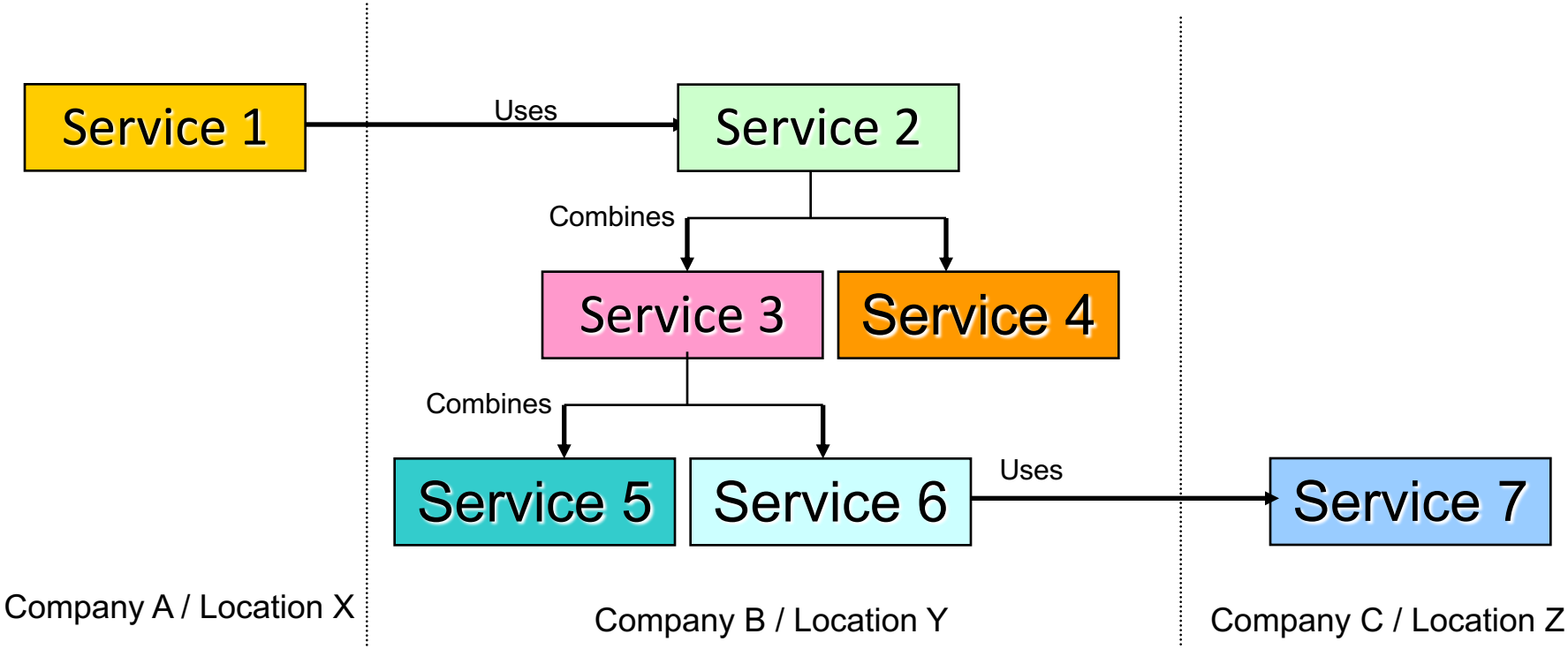
From “Enterprise SOA”, D. Krafzig, K.Banke, D.Slama., Prentice Hall, 2005

Coupling

	Tight Coupling	Loose Coupling
Interface	Class and Methods	REST like (i.e. fixed verbs)
Messaging	Procedure Call	Document Passing
Typing	Static	Dynamic
Synchronization	Synchronous	Asynchronous
References	Named	Queried
Ontology (Interpretation)	By Prior Agreement	Self Describing (On The Fly)
Schema	Grammar Based	Pattern Based
Communication	Point to Point	Multicast
Interaction	Direct	Brokered
Evaluation (Sequencing)	Eager	Lazy
Motivation	Correctness, Efficiency	Adaptability, Interoperability
Behaviour	Planned	Reactive
Coordination	Centrally Managed	Distributed
Contracts	By Prior Agreements, Implicit	Self Describing, Explicit

Extensibility via Layered Services

- Each layer defines a communication service, and each service has a clearly defined Interface with the layer above and below.
- As long as the contract (interface) between services remains the same, the services can be changed easily.



Enterprise Service Bus

A new form of *Enterprise Service Bus* (ESB) infrastructure – combining message-oriented middleware, web services, transformation and routing intelligence – will be running in the majority of enterprises by 2005.

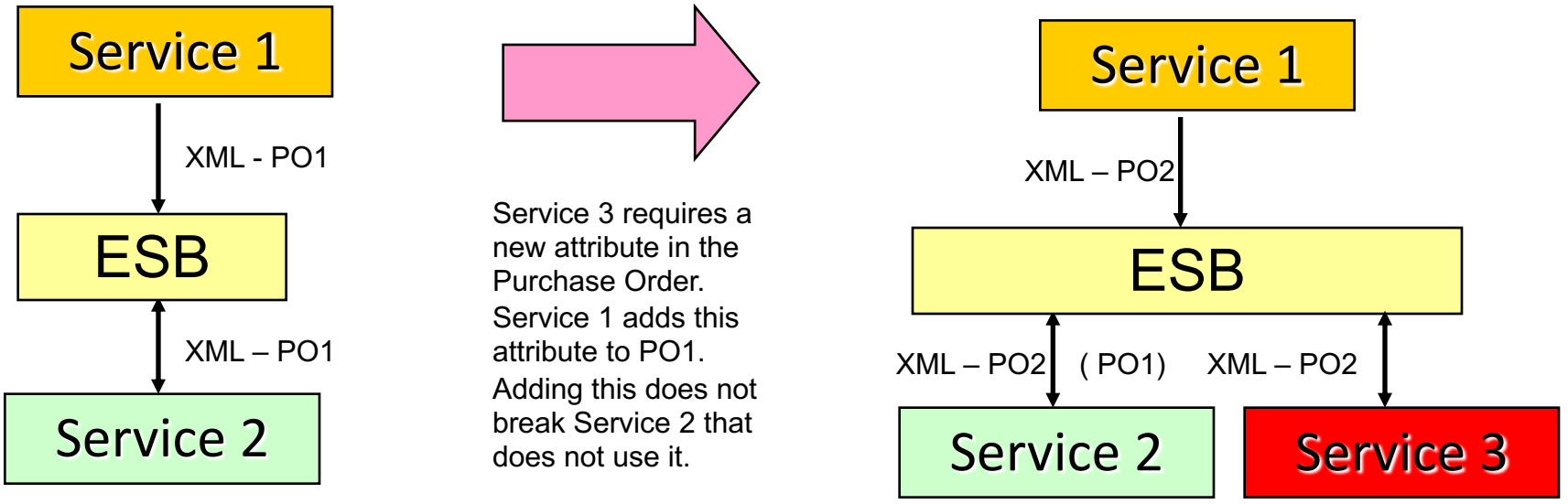
These high-function, low-cost ESBs are well suited to be the backbone for service-oriented architectures and the *enterprise nervous system*."

Gartner, 2002. DF-18-7304.

XML and Loosely Coupled

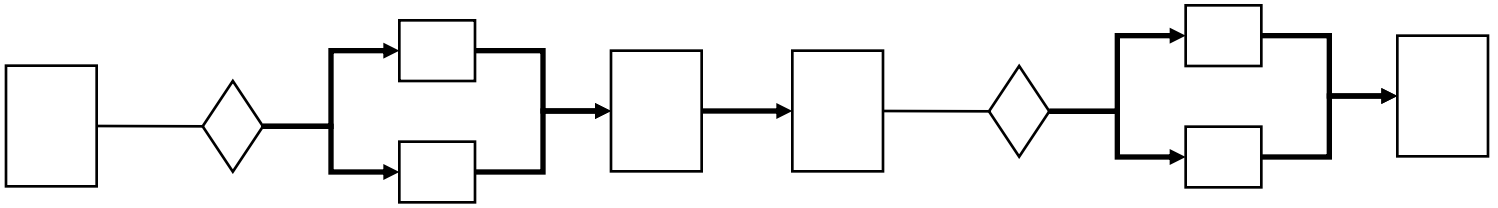
- Loosely Coupled :
- Allows Component/Service flexibility and choice.
- Contract = implementation at interface not implementation.
- Document driven process - using XML standard.
- Process is robust to changes in XML documents.

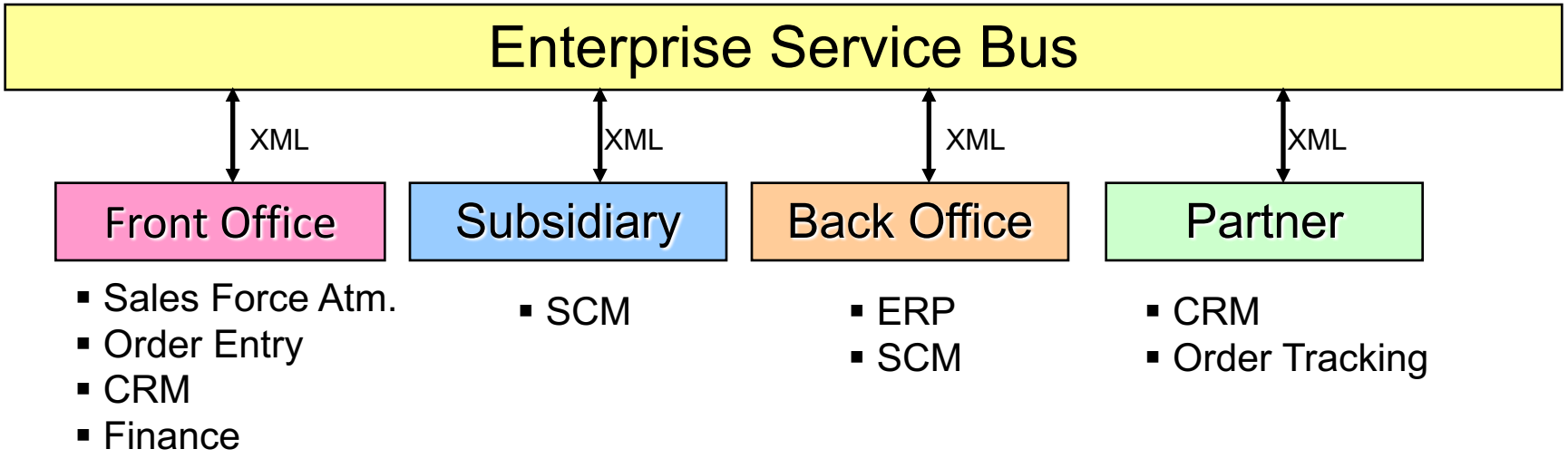
Service 1 sends a Purchase Order (PO) to Service 2.



Integrating Process Flow

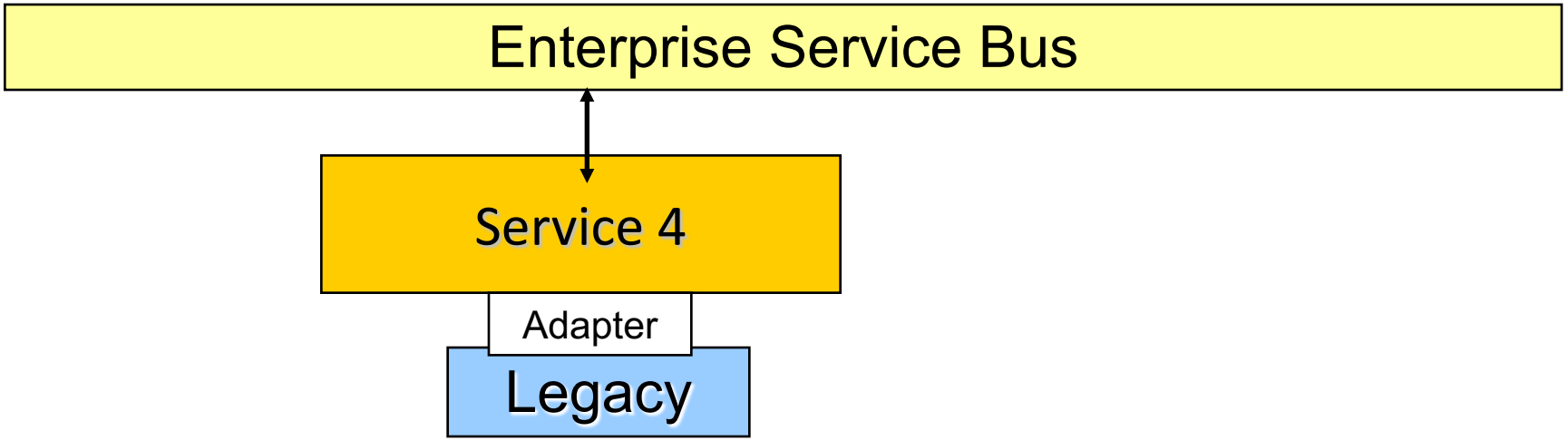
An ESB process capabilities range from simple sequencing to sophisticated business process orchestration, using for example, BPEL4WS





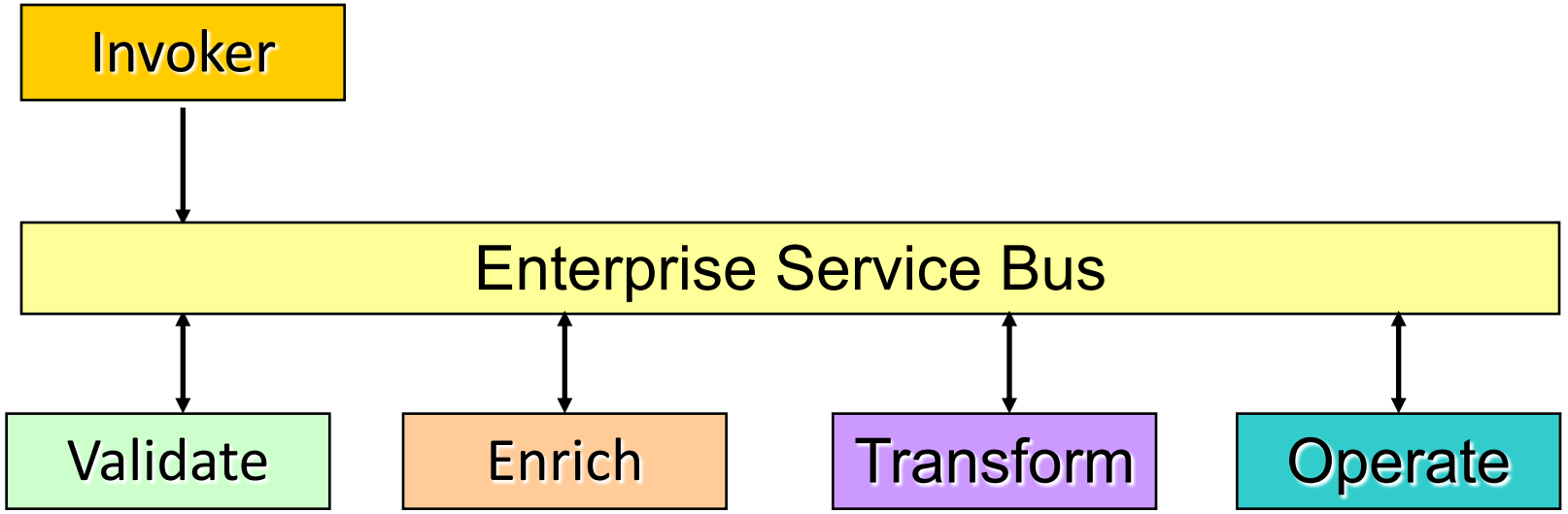
Integrating Legacy Systems

- Reuse legacy systems, don't need to re-write them.
- Wrap the legacy system and expose XML API's as Service



VETO Pattern (Dave Chappell)

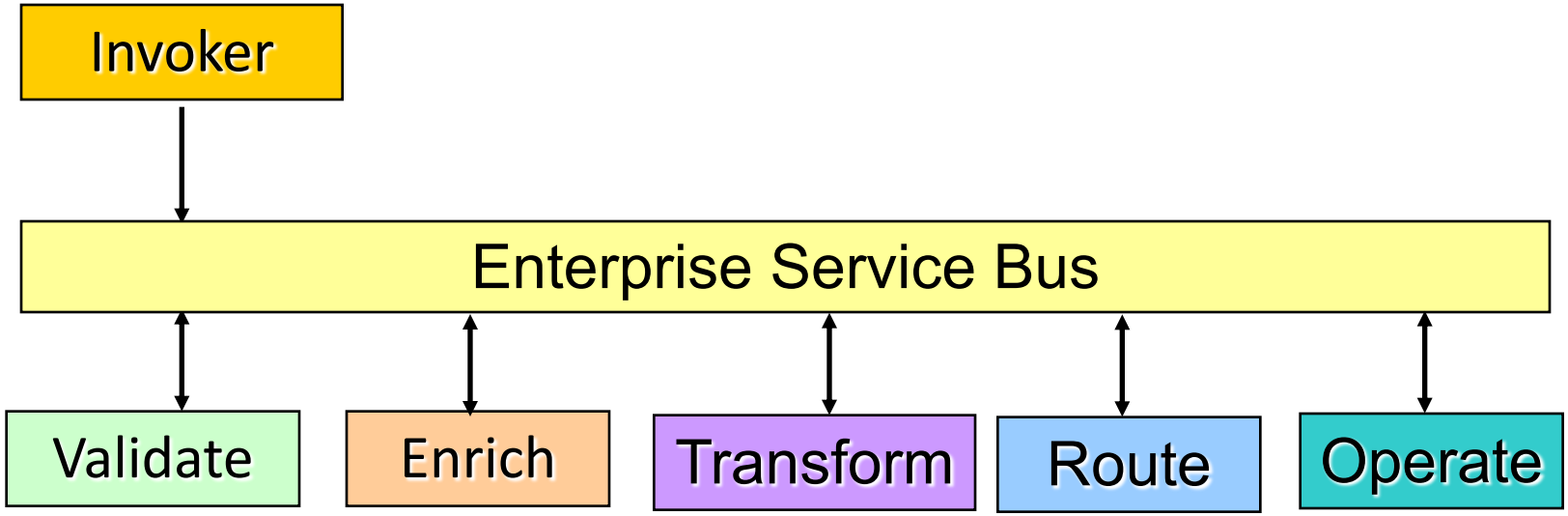
- Validate :- check XML, WSDL, use XPath to check details
- Enrich :- add additional data to the message
- Transform :- convert message to target format , using XSLT
- Operate :- invoke the target service



VETRO Pattern (Dave Chappell)

Add in one step:

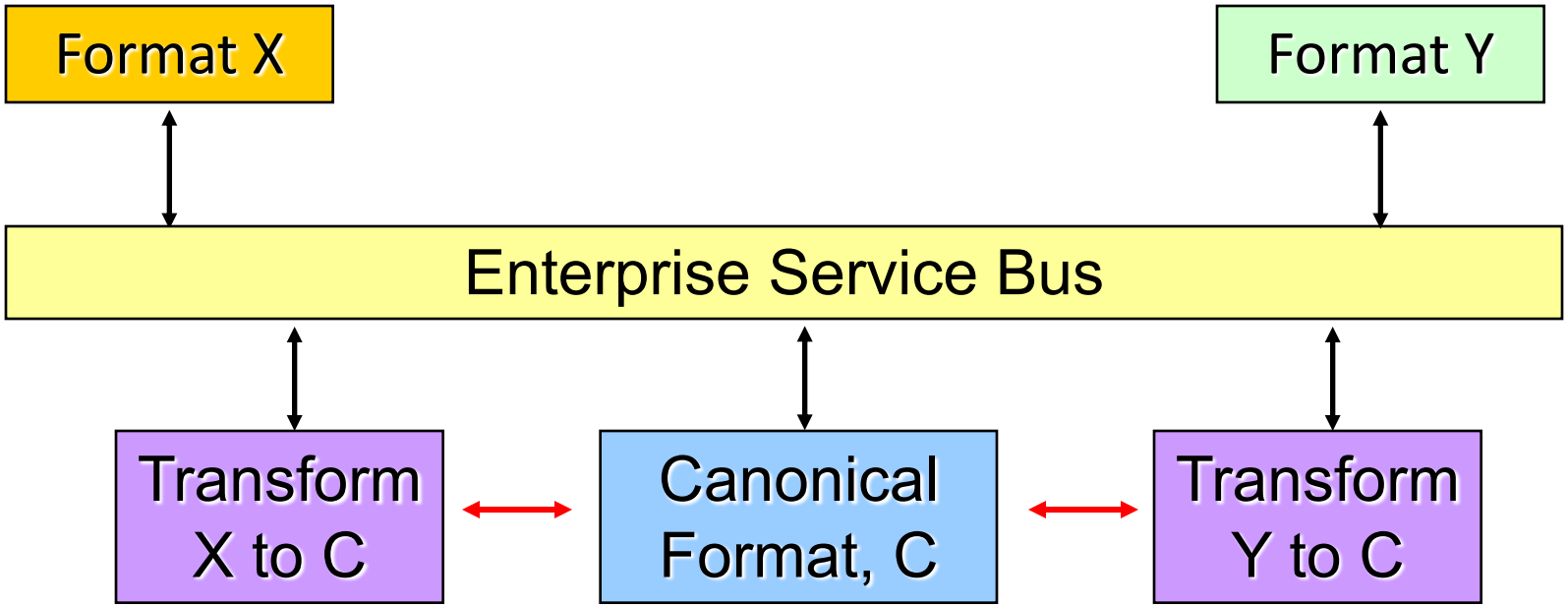
- Route :- send the message to a destination based on content (content based routing) business process, workflow or itinerary.



XML - Canonical Message Format

In the middle:

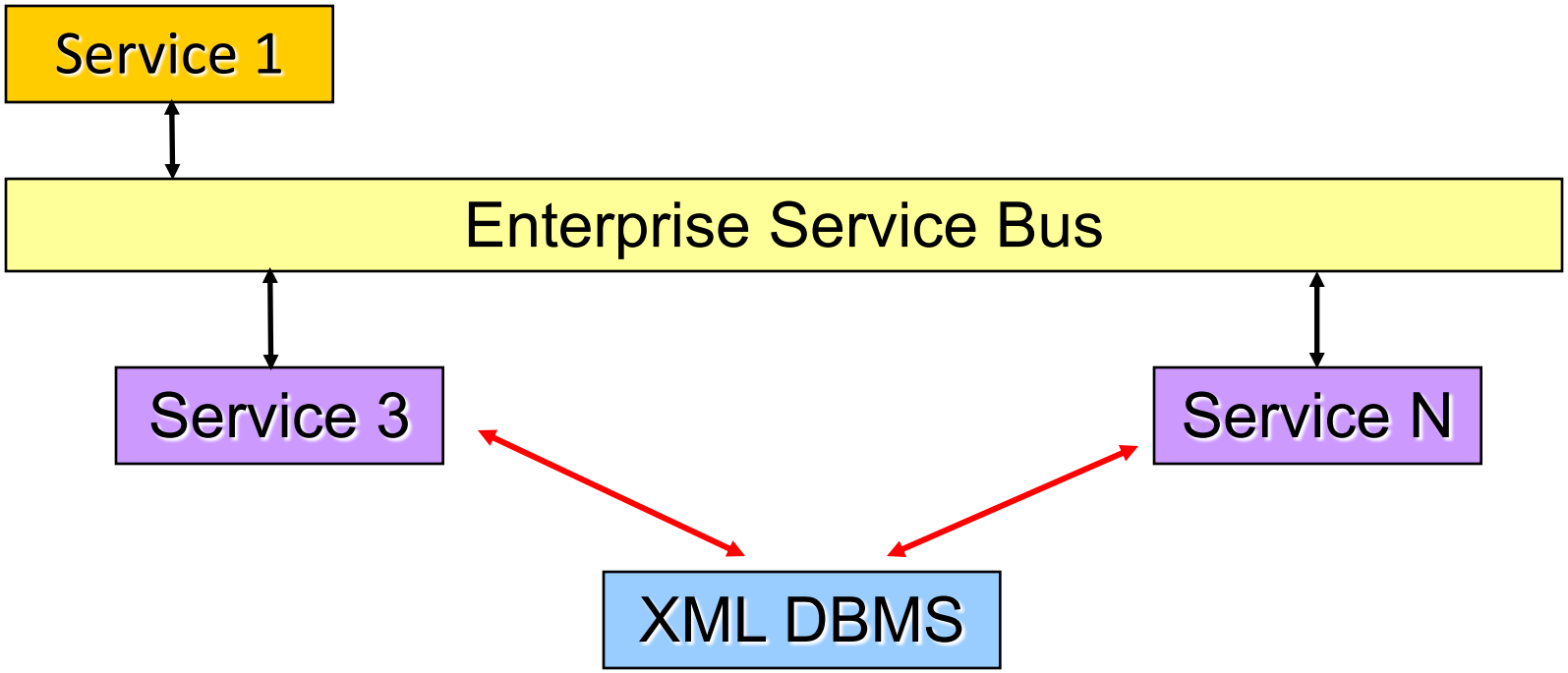
- An application independent version of the data;
- Services translate in/out to this format;
- Based on Enterprise Data Model or Industry Standard.



XML - Shared Persistence

A shared XML Persistence Service:

- Optimises multiple operations on same data, avoids multiple parse, serialise, parse operations.



Transactions

- Use Case: Money Transfer by a debit and then credit services.
- Need to ensure both succeeds.

WS-I Basic Profile 1.0

- SOAP 1.1
- WSDL 1.1
- UDDI 2.0
- XML 1.0 (Second Edition)
- XML Schema Part 1: Structures
- XML Schema Part 2: Datatypes
- RFC2246: The Transport Layer Security Protocol Version 1.0
- RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile
- RFC2616: HyperText Transfer Protocol 1.1
- RFC2818: HTTP over TLS
- RFC2965: HTTP State Management Mechanism
- The Secure Sockets Layer Protocol Version 3.0

WS-* Coalition: BEA, IBM and Microsoft

- WS-Addressing
- WS-Attachments
- WS-BusinessActivity
- WS-Choreography
- WS-Coordination
- WS-Discovery
- WS-Enumeration
- WS-Eventing
- WS-Federation
- WS-Inspection
- WS-Manageability
- WS-MetadataExchange
- WS-Notification
- WS-PolicyFramework
- WS-Provisioning
- WS-ReliableMessaging
- WS-Resource
- WS-Security
- WS-Topics
- WS-Transactions (AtomicTransactions)
- WS-Transfer

Two Main Types of Integration

- **Internal (A2A) :** Systems communicate within organisation, To:
  - Re-organise after a merger;
  - Automate business process;
  - Consolidate data entry across disparate systems;
  - Re-use information and system assets (services);
  - Regulatory Compliance;
  - Single Customer view.
- **External (B2B):** Systems communicate across organisations, To:
  - e-Business, e-Commerce, e-Procurement, e-etc;
  - Extend business process outside company;
  - Supply chain;
  - Outsource parts of business.

Levels at which Integration is done

- Presentation:
  - Screen Scraping;
  - Remapping legacy interfaces across applications to a new common GUI;
- Portals – display information retrieved from multiple applications via unified GUI, e.g. portlets, collaboration applications;
- Data – at the logical data base layer, using data transfer or sharing;
- Functional – at the logical business level using application code (integration tier), JCA, distributed objects, SOA, middleware;
- Process – long running processes than span multiple applications, e.g. BPEL, BPMS.

Types of Integration Technology

- Presentation Based:
  - Portals
  - Collaboration
- Data (DBMS) Based
- Files, FTP, Connect Direct
- Method Oriented (API), e.g. JD Edwards, PeopleSoft, Baan, SAP, Oracle.
- RPC, e.g. Corba, COM
- Transaction Monitors, e.g. CICS, Tuxedo,
- Message Oriented, e.g. MQ, MSMQ, JMS
- Application Server e.g. J2EE, .Net + Server 2003
- EAI brokers
- Service Oriented Architecture ← Focus Today
  - Web Services
  - Enterprise Service Bus

Web Services Architecture

Web Services are one popular way to implement SOA.

"A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."

W3C

Gartner Predicts 2005: The Impact of Web Services Still Grows.

SOA Architecture Principles

A SOA will need to adhere to several overarching principles:

- Service-based, not just code-based
- Assembled, not built; new term:"Composite Applications"
- Physically distributed
- Manageable
- Implemented in layers
- Logically tiered for separation of concerns
- Loosely Coupled
- Standards Based

Service Oriented Architecture (SOA)

- Services (not applications) are the building blocks of the Enterprise;
- Provides an enterprise wide standard;
- Involves creating standardised interface technology so that when a service is deployed, it is available for use by other services;
- Standardisation allows legacy silos to be easily accessed, integrated and aggregated, resulting in an enterprise wide tier of business services;
- SOA is an Architectural Style

Four Tenets of SOA

Don Box's (Microsoft) 4 Tenets of SOA:

1) **Boundaries are explicit.**
2) **Services are autonomous.**
3) **Services share schema and contract, not class.**
4) **Service compatibility is determined based on policy.**

Choreography Vs Orchestration

•The difference between them is subtle and confusing – and many standards disagree.

•**Orchestration** – Business process extends outside of business and involves multiple parties but is controlled by the business.

•WS-BPEL : Web Service Business Process Execution Language

•BPEL4WS :

•**Choreography** – Pre-defined plan; participants act independently, in a more collaborative manner. Tracks the sequences of messages involving multiple parties and sources.

•WS-CDL : Web Service Choreography Description Language

•WSCI : Web Service Choreography Interface

ESB Vendors (claimed on web page)

- Sun ICAN – (SeeBeyond);
- Cape Clear;
- Fiorano;
- IBM WebSphere ESB
- BEA AquaLogic;
- Iona;
- Oracle Fusion
- Kenamea
- Microsoft WCF (Indigo)
- KnowNow;
- PolarLake;
- SnapBridge Software;
- Sonic Software;
- SpiritSoft;
- Wakesoft;
- webMethods.
- Software AG

Movements in the market place

- Integration Broker Players:
- SeeBeyond
- webMethods
- IBM
- TIBCO
- BEA
- Vitria

- Movements:
- B2B to EAI
- B2B add EAI
- EDI to B2B
- B2B to ESB
- EAI to ESB
- BPM to ESB

Change is rapid: This slide will always be out of date…….

ESB Open Source

- Apache Synapse
- Service Mix
- etc

- Bla bla

ESB fixes many Integration Broker problems:

- **Drawback of Integration Brokers is the Hub-Spoke Architecture:**
  - Topology doesn't allow regional control over local domains
  - BPM cannot span business units and businesses.
  - Limited by MOM to cross different physical networks, network segments, firewalls.

- **ESB adds to Integration Broker concepts:**
  - Adopt XML
  - SOA
  - Web Services: Integration can span business units and businesses.
  - Reliable Asynchronous Messaging
  - Separate Process Logic from Interface Logic. Choreography can span business units and businesses

SOA Performance Management Issues

SOA based systems span infrastructure tiers, software, middleware, operating systems and corporate and national boundaries. Management Tools must:

- Works across multiple vendor's hardware and software stacks;
- Monitor and correlate information about transactions across multiple tiers.
- Help IT specialists from different fields drill down to find cause of performance problems;
- Provide Troubleshooting advice and guidance.

SOA Governance

"The art and discipline of applying structured relationships, procedures, and policies to produce managed outcomes for services consistent with measurable preconditions and expectations."

Zapthink, 2007

- Service Quality (fitness for use) must be proven; consumers need to know what quality is.
- Tightly coupled systems define governance and control in context of application – not possible with SOA.
- Need a Service contract – based on Policy
- ********** to finish look at Systinet PDF
- Compliance

SOA Methodologies

"By 2007, 70 percent of SOA and Web services engagements will require a 'cohesive, end-to-end service delivery methodology and tool set' from an ESP for end users to maximize their investments (0.9 probability)."

Gartner

Michele Cantara, "Common Features of External Service Providers' SOA Frameworks and Offerings," September 2005

SOA Methodologies

- IBM SOAD : Service Oriented Analysis & Design
- IBM SOMA : Service Oriented Modeling & Architecture
- Microsoft MOTION
- Thomas Earl's Approach.
- Sun RQ - based on business use case driven approach and UP

How do you model a service ?

- **Top Down:** Start from business process and break it down to find tasks that can be re-used across the enterprise.
- **Bottom Up:** Legacy systems provide service units. Currently independent components (tasks) that can be used across applications, across business units.
- **Left-Overs:** “Things” that are not natural domain objects can be modeled as a service. e.g. Account Transfer is not naturally part of a Bank Account but something else.

- Decision is based on Principles described earlier.

6 Roles in Domain-Driven Design

- **Entities** : Objects with a distinct identity, includes
- **Value objects** : Objects with no distinct identity
- **Aggregates** : car = wheels + doors+ seats ….
- **Factories** : Define methods for creating entities
- **Repositories** : Manage collections of entities and encapsulate the persistence framework
- **Services**: Implement responsibilities that can't be assigned to a single class and encapsulate the domain model.
  - Allows a use case to interact with things above;
  - May have methods for each use case step;
  - Wraps a business transaction

From "POJOSs in Action" Eric Evans 2003

SOAP Styles and Uses

Two possible *styles*:

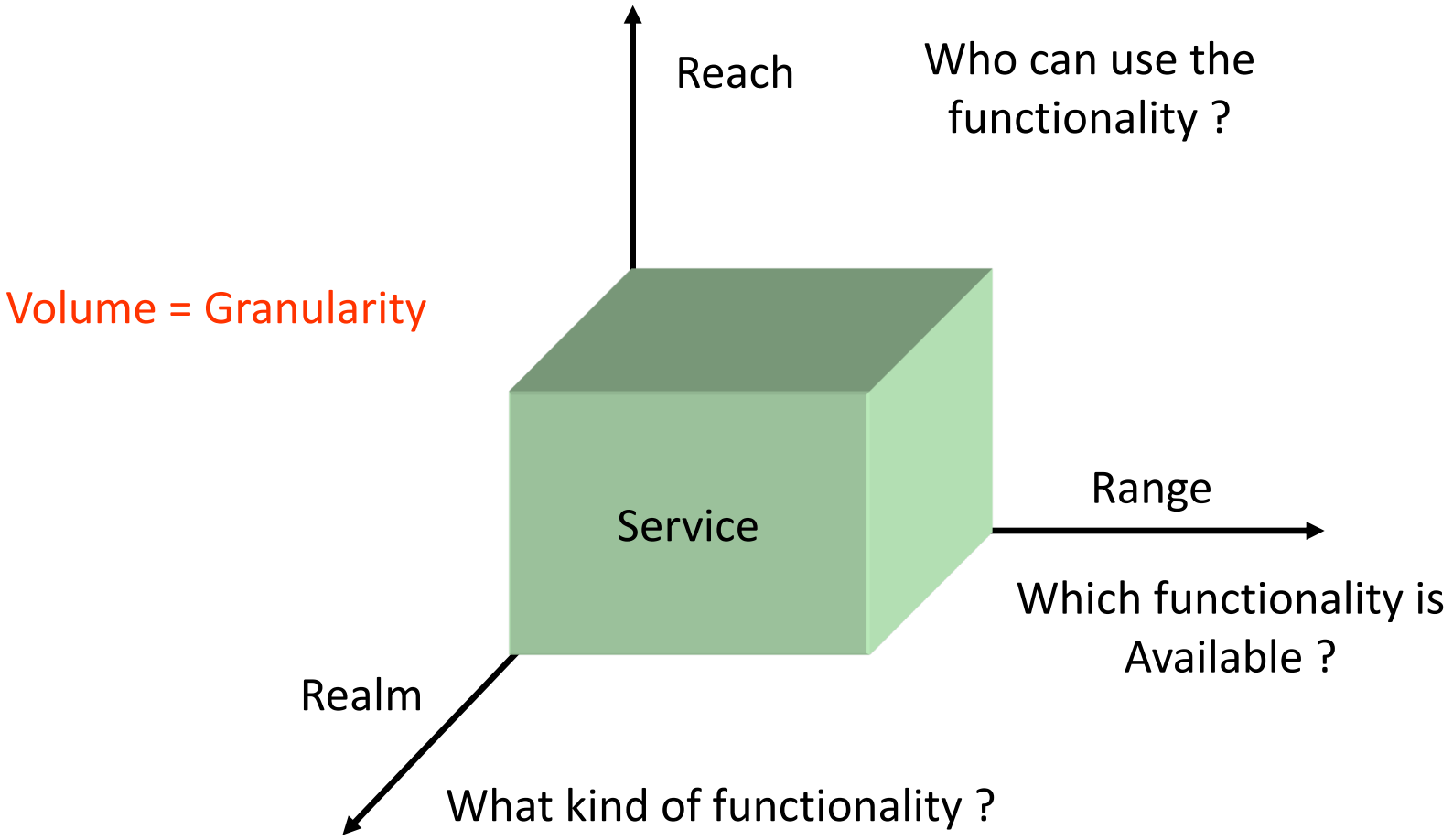
- ***RPC***. Implies a SOAP body structure which indicates service name, and multiple parameters and return values
- ***Document***. Implies a SOAP body which is a complex message document

Two possible *uses*:

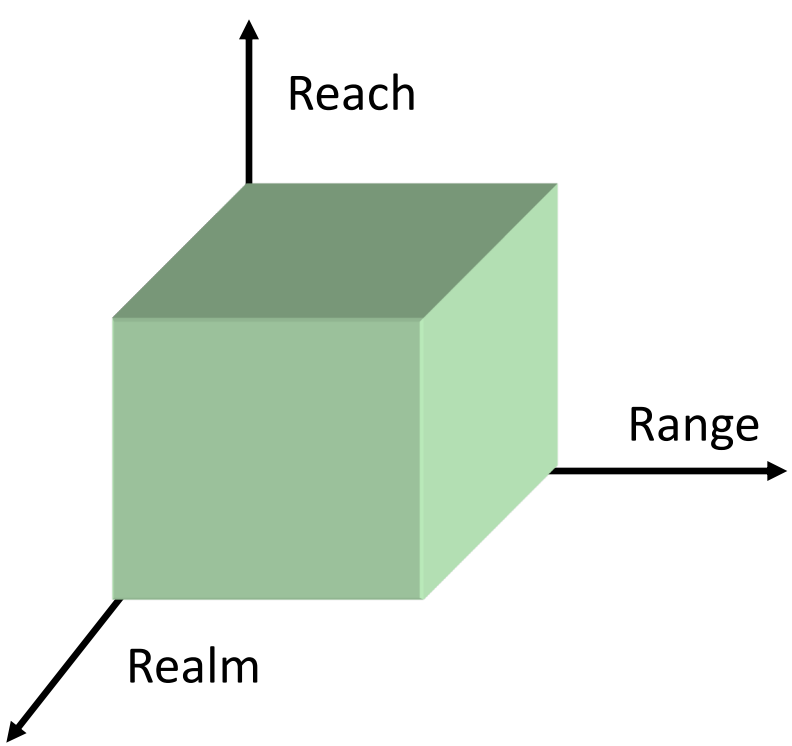
- ***Encoded***. Adheres to a set of rules for serialising a graph of typed objects using basic XML schema data-types, but as a whole, does not conform to a schema
- ***Literal***. Body content conforms to a specific XML schema

Best approach today is **Document / Literal.**
RPC / Encoded is no longer supported.

Granularity: 3D Model of Service

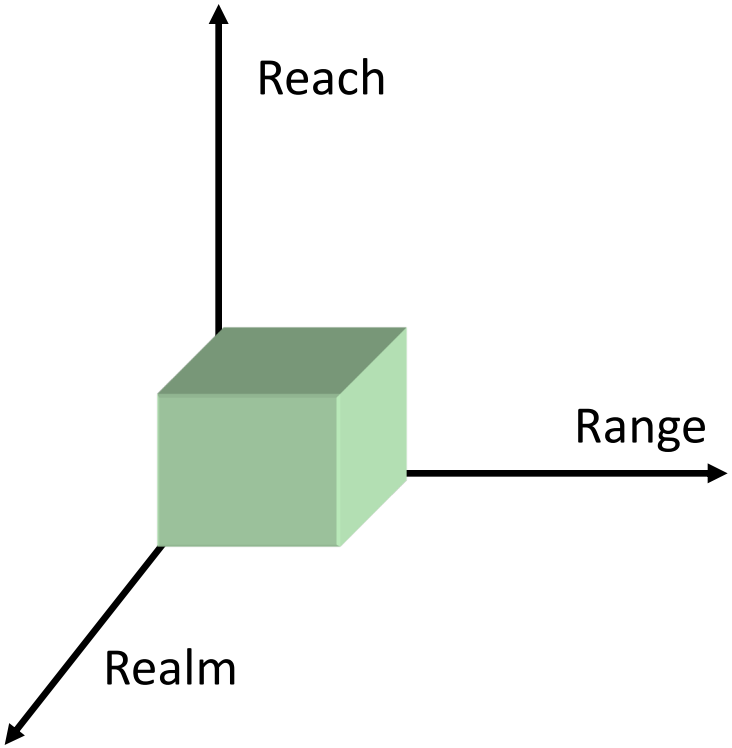


Granularity = Volume



Coarse Grained Service

High Abstraction

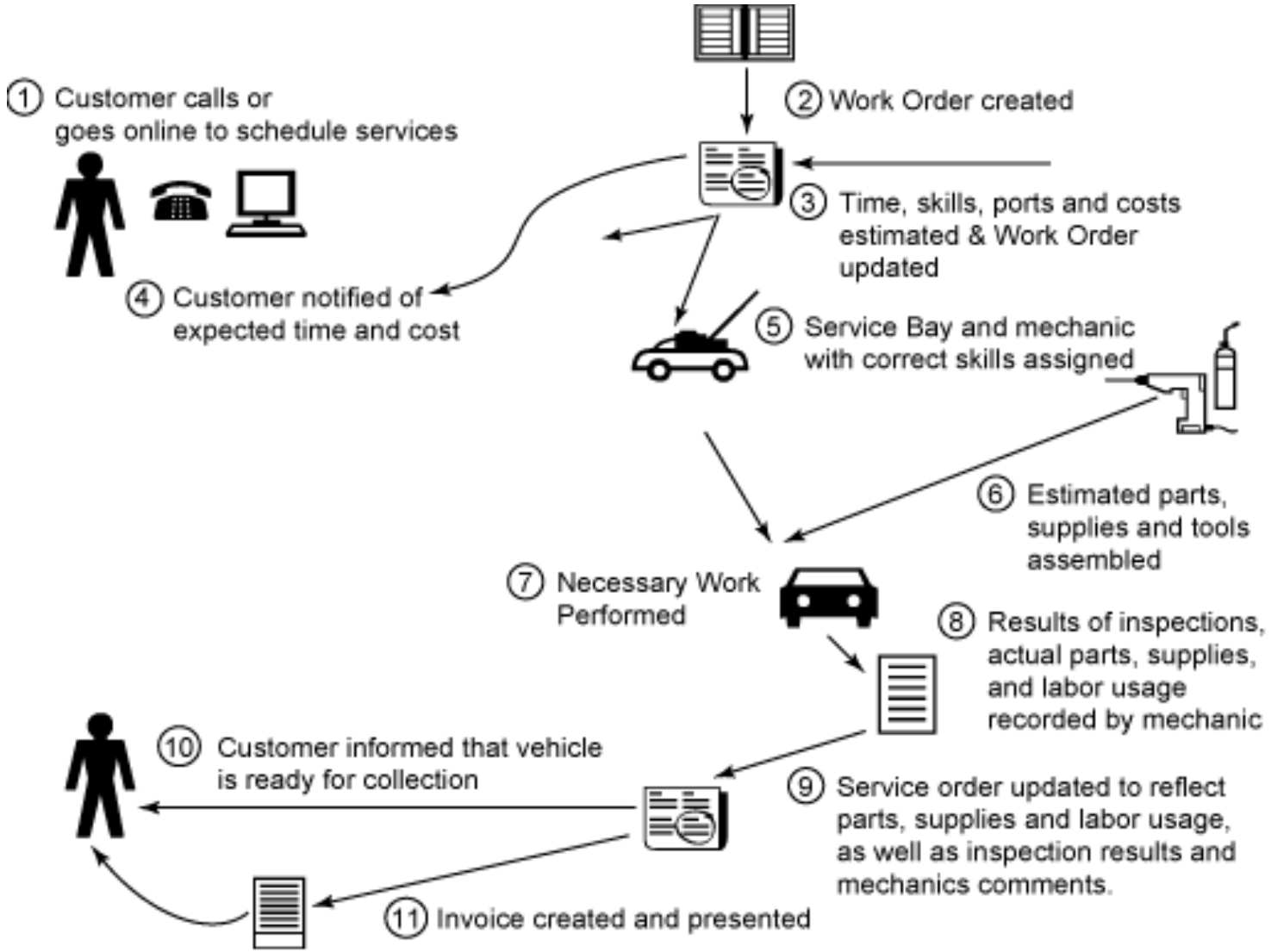


Fine Grained Service

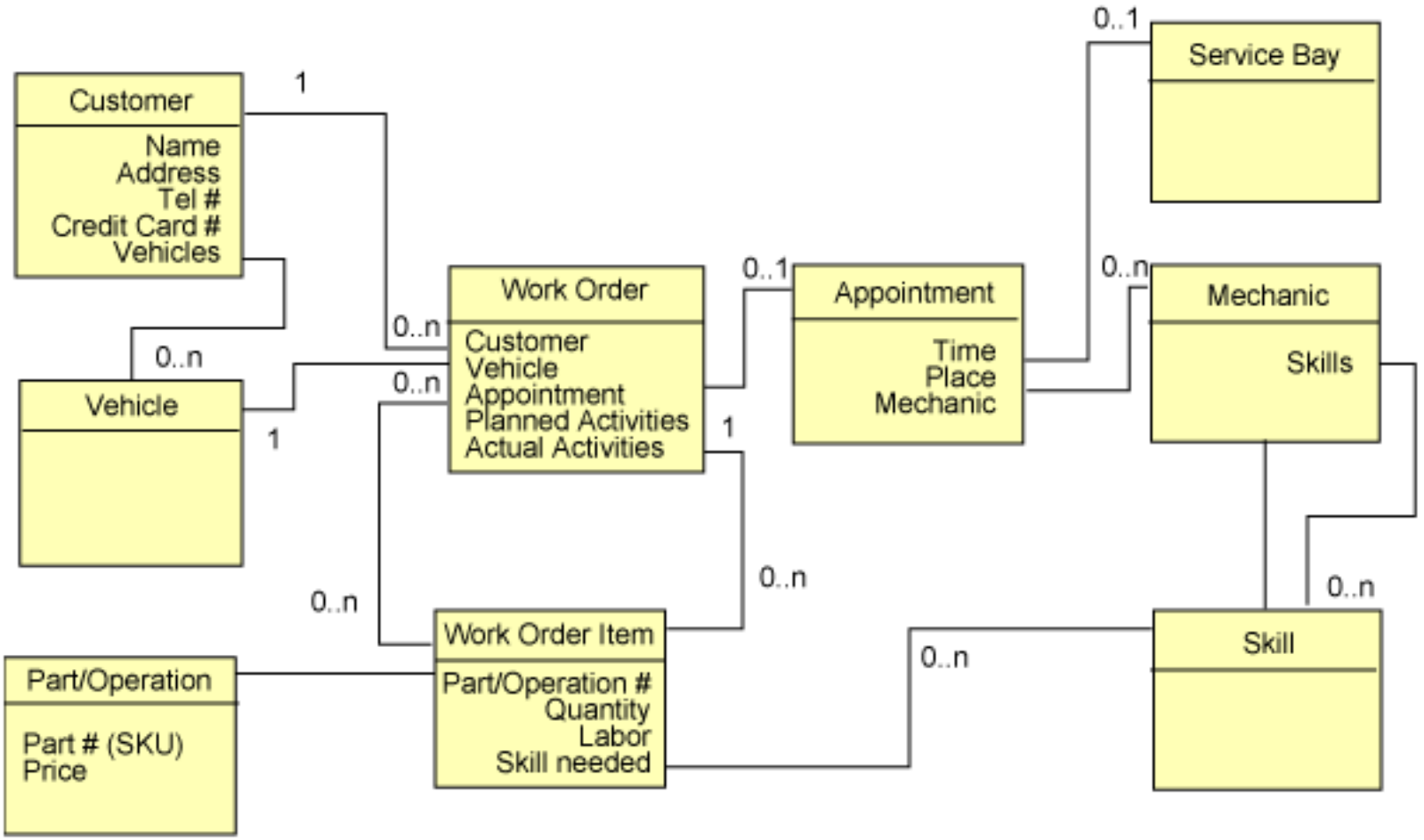
Low Abstraction

Complexity of Service increases as volume increases.

Example Background Use Case



An OO Example



Example Services

Work Order Service

Operations:
- Creative Work Order
- Operation: Create Work Order Item
- Calculate Estimated or Actual Costs
- Update Work Order Item

Scheduling Service

Operations:
- Schedule skilled mechanic and service bay

Customer Service

Operations:
- Lookup customer by tel #
- Create new Customer
- Get Customer Vehicles

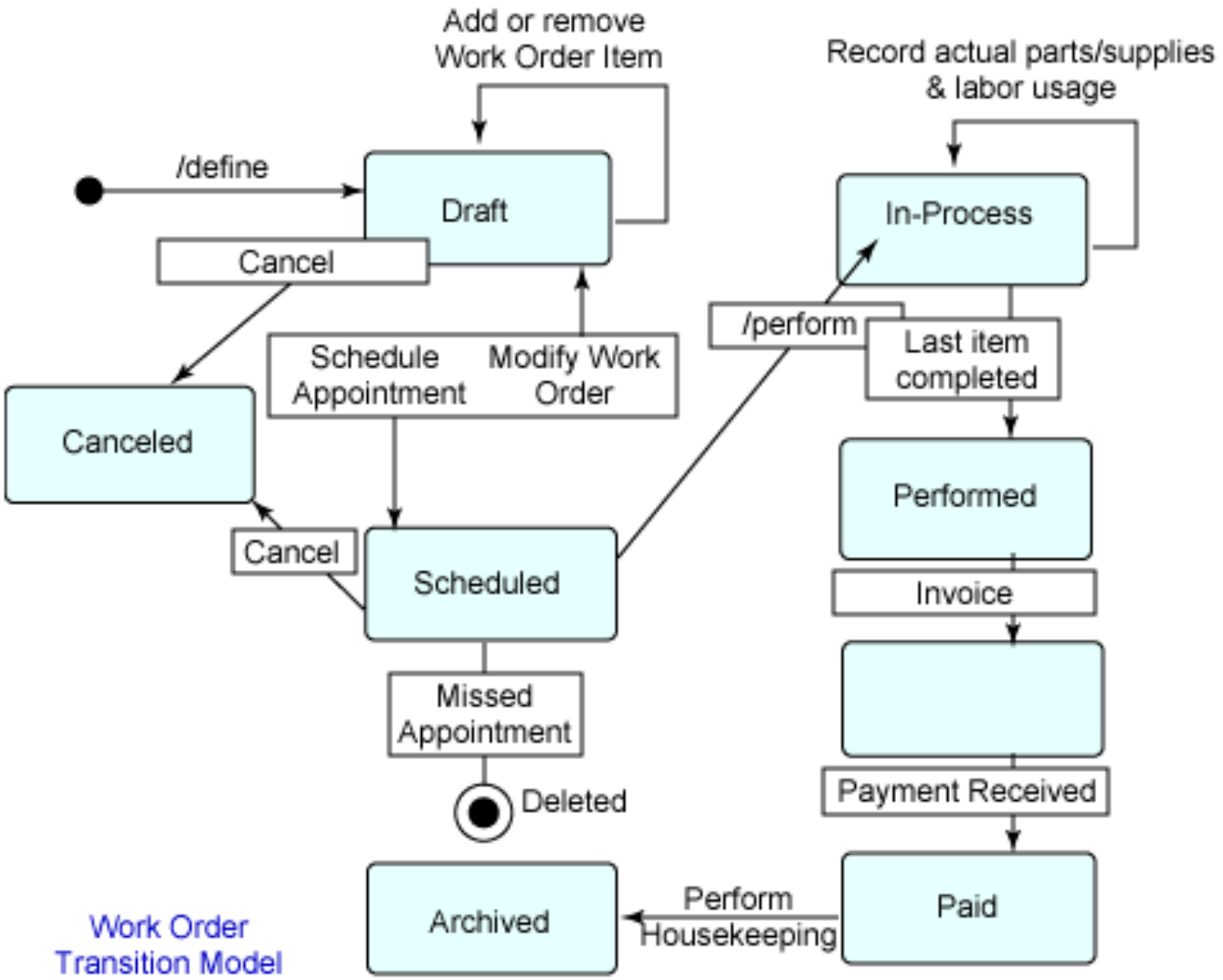
Catalog Service

Operations:
- Lookup/List available maintenance offerings by vehicle.
- List parts, supplies and labor for a specific maintenance item
- Lookup Parts and Supplies
- List valid alternative parts/supplies

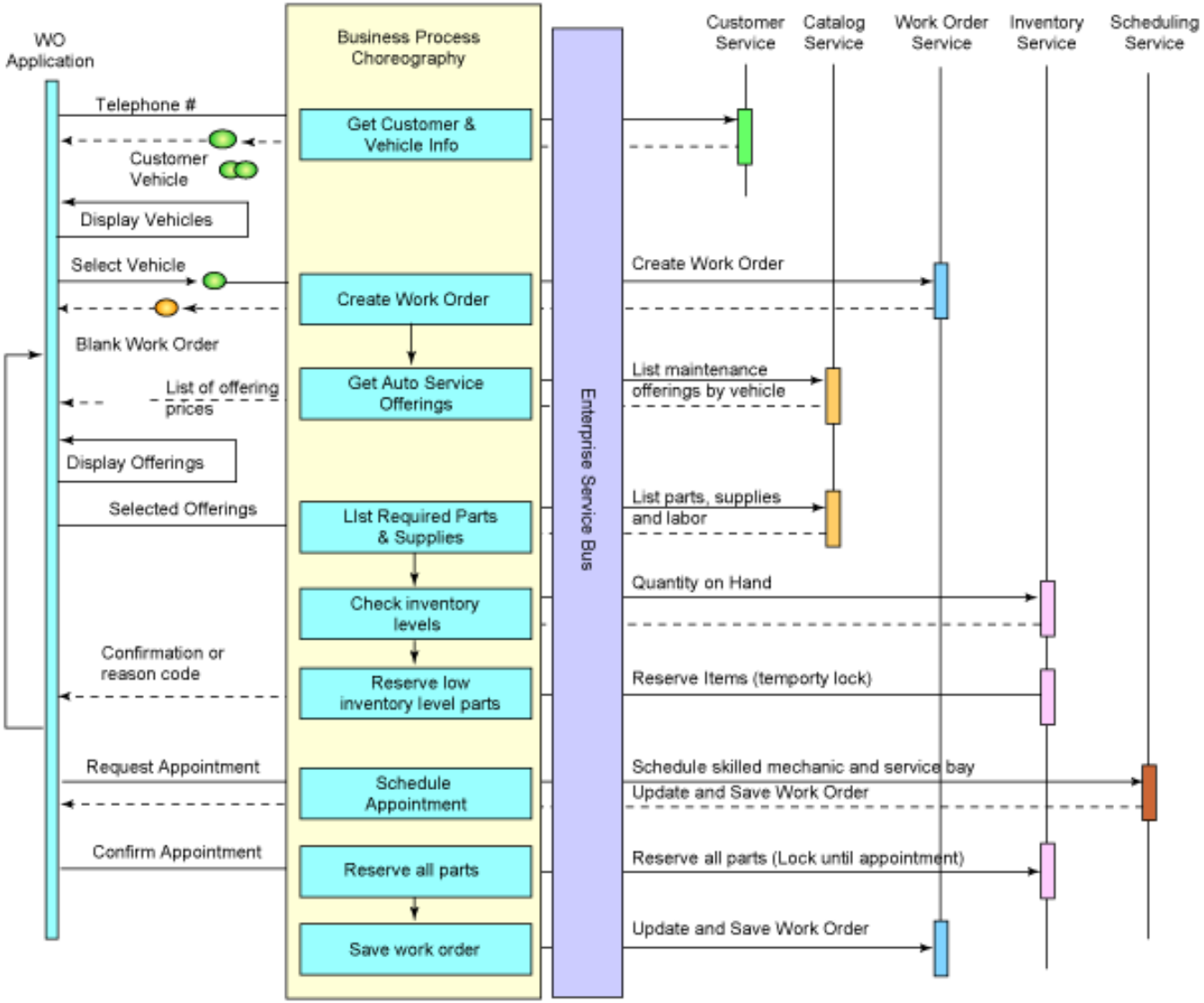
Inventory Service

Operations:
- Determine Quantity on Hand of Item
- Special Order Item
- Backorder Item
- Expected Arrival Date for Part

BPM for Work Order



Business Interaction Model for Work Order



SOA Infrastructure Justified

SOA Scope and Complexity	Low Organisation Size and Complexity	High Organisation Size and Complexity
High	**High SOA Complexity Small Enterprise** Need for robust SOA infrastructure will depend on whether the investment is justified by business requirements.	**High SOA Complexity Large Complex Enterprise** An investment in a robust SOA infrastructure solution may be justified by business requirements.
Low	**Low SOA Complexity Small Enterprise** It is unlikely that an investment in a robust SOA infrastructure would justified by business requirements.	**Low SOA Complexity Large Complex Enterprise** An investment in a robust SOA infrastructure solution is probably not justified by business requirements at this time, maybe in future.