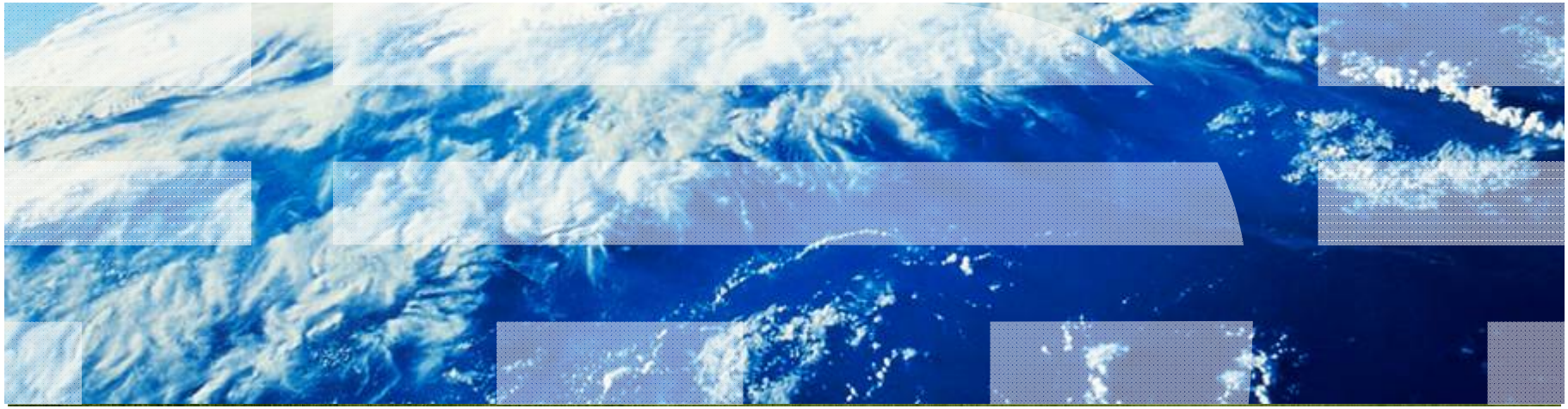
Dr. Hans-Peter Hoidn
Executive Architect, IBM
Distinguished IT Architect (Opengroup)

University of Zurich IBM

Enterprise IT Architectures

SOA (Service Oriented Architecture)



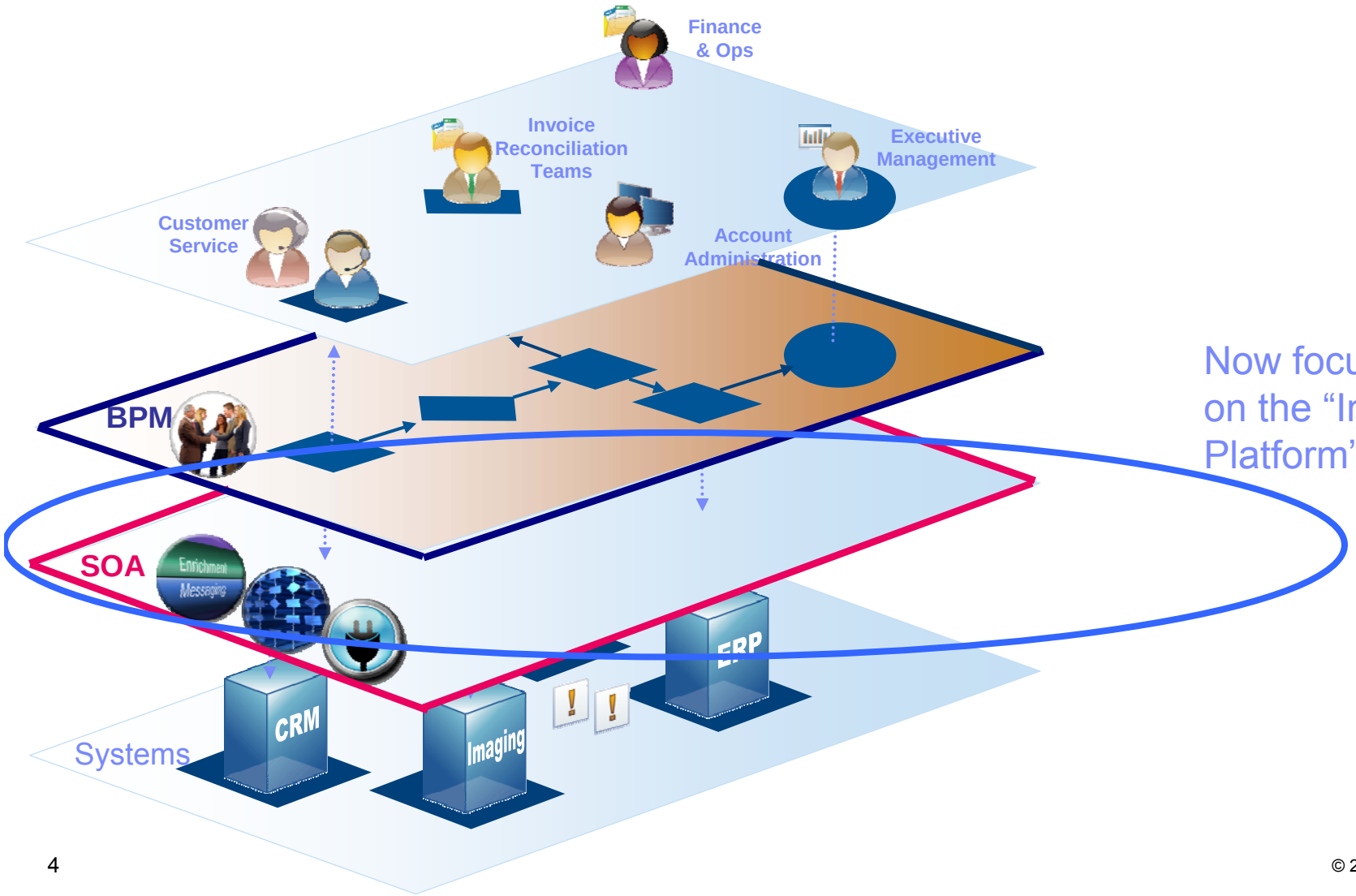
SOA Introduction

Agenda of this Session

- **Last Session:**
  - **Introduction of BPM (Business Process Management)**
  - **Focus on Capturing processes and automated execution**

- **This Session**
  - **BPM needs to be augmented by integration to applications, components, services, data bases etc.**
  - **SOA provides the mechanisms to do integration ("*Integration Platform*") such that both sides of an integration are independent**
  - **Methodology to structure distributed applications including business processes as well as user interfaces**

Positioning of SOA



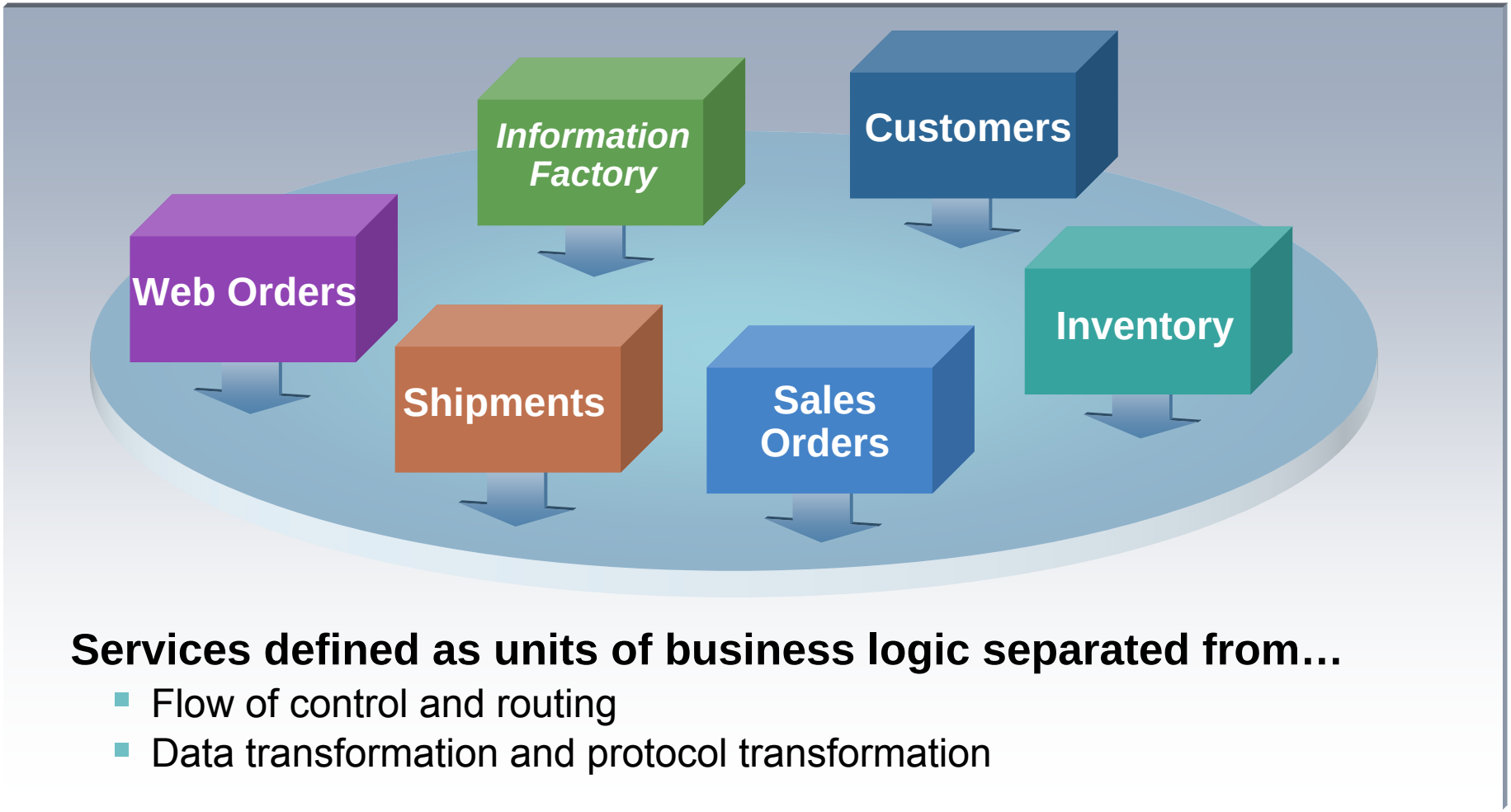
Now focusing on the "Integration Platform"

What is SOA

- **SOA is an *architectural style* or approach whose goal is to achieve loose coupling among interacting software agents**
- **All functions (that need to be used by more than one system) are defined as "*services*"**
- **Service providers agree to a defined, implementation-independent interface with service clients**
- **Services oriented architecture is the *policies, practices and frameworks***
  - **that enable application functionality and IT services to be**
  - **provided and requested as a set of services**
  - **using a standards based form of interface.**

Service Oriented Architecture

Moves IT Logic Out of Services

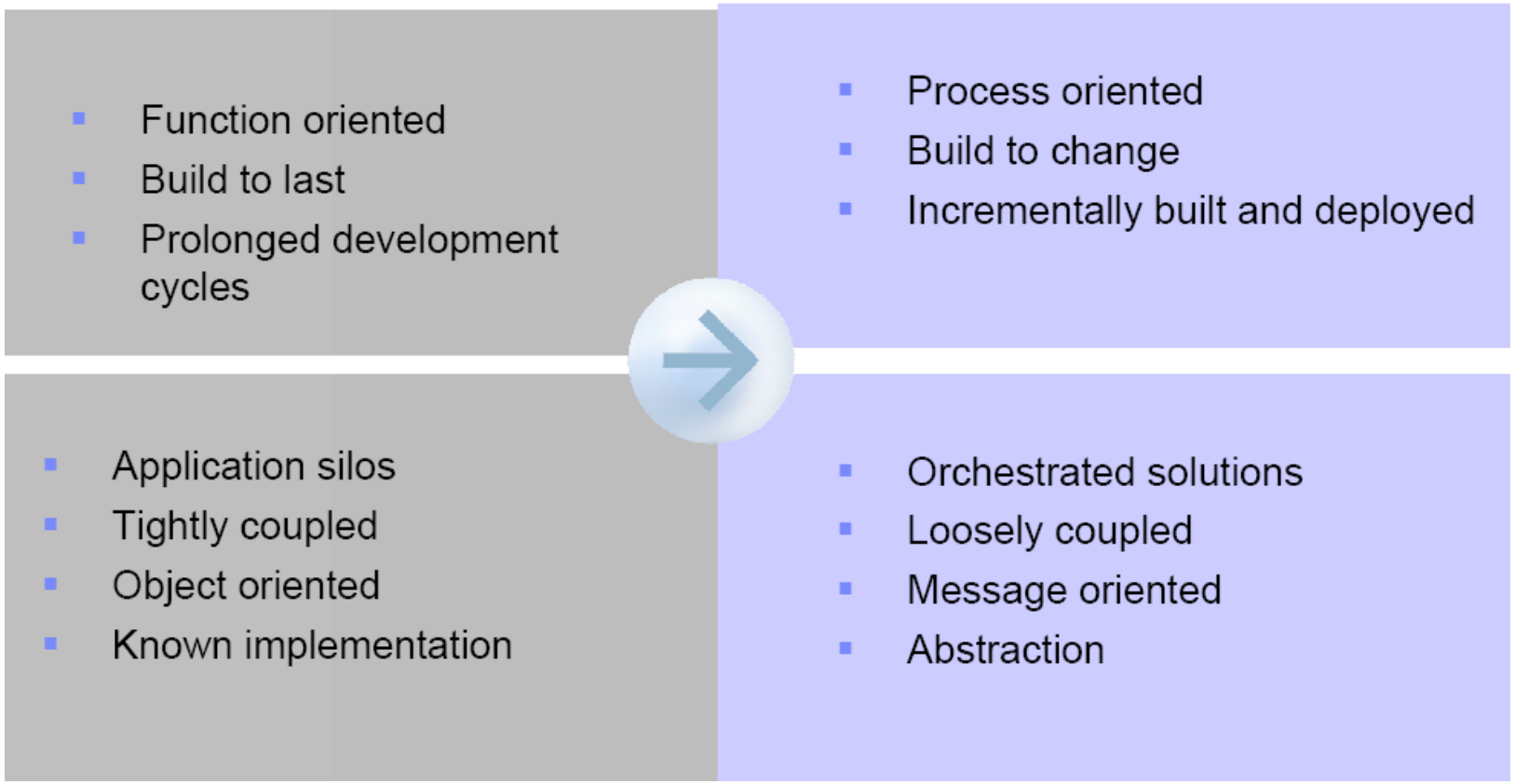


Services defined as units of business logic separated from…

- Flow of control and routing
- Data transformation and protocol transformation

SOA addressing IT as well as Business – common shift

Shift to a Service-Oriented Architecture



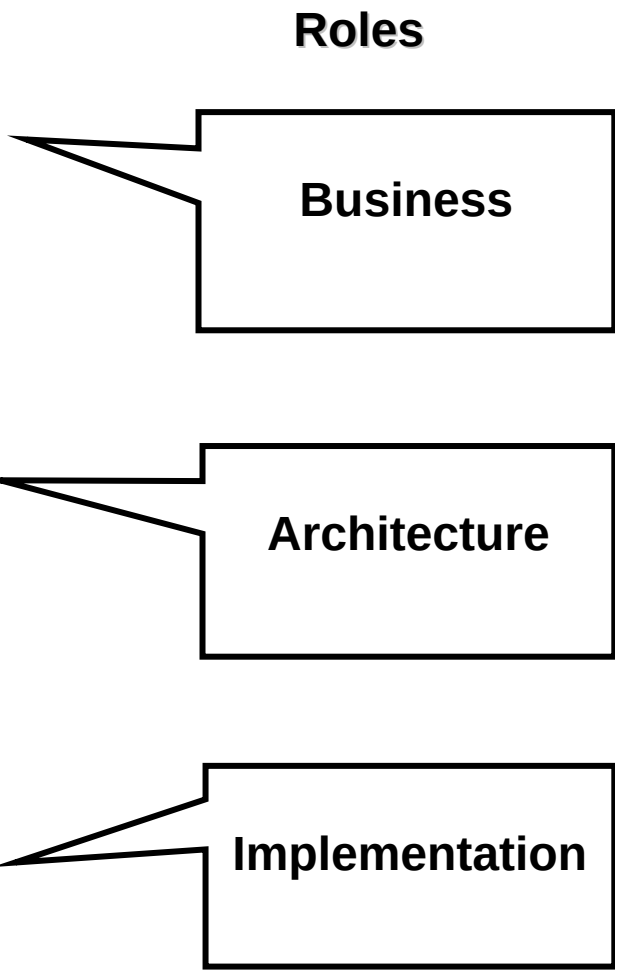
SOA is different things to different people

A set of services that a business wants to expose to customers and clients

an architectural style which requires a service provider, requestor and a service description.

a set of architectural principles and patterns which address characteristics such as *modularity, encapsulation, loose coupling, separation of concerns, reuse, composable and single implementation*.

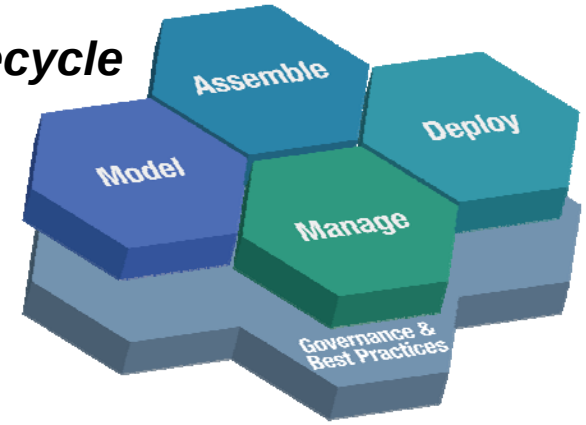
A programming model complete with standards, tools, methods and technologies such as web services.



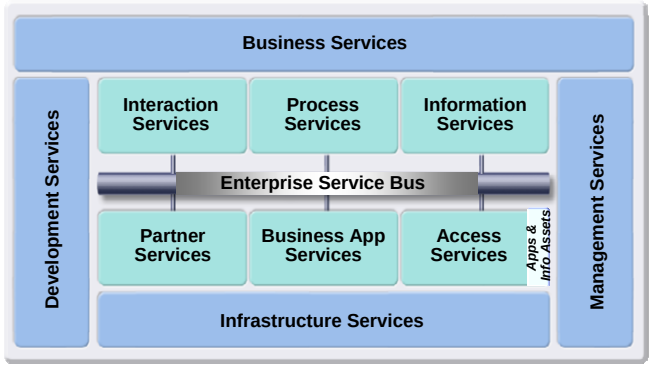
SOA Key Concepts

Key Models and Methods for SOA – Enabling greater flexibility in Enterprise IT Architectures

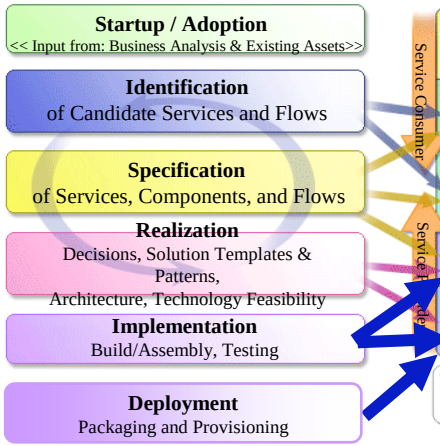
SOA Lifecycle



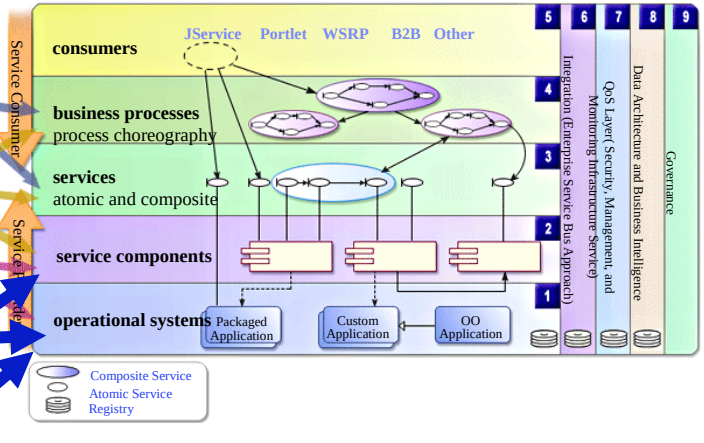
SOA Reference Architecture



The SOMA Method: Service-Oriented Modeling and Architecture

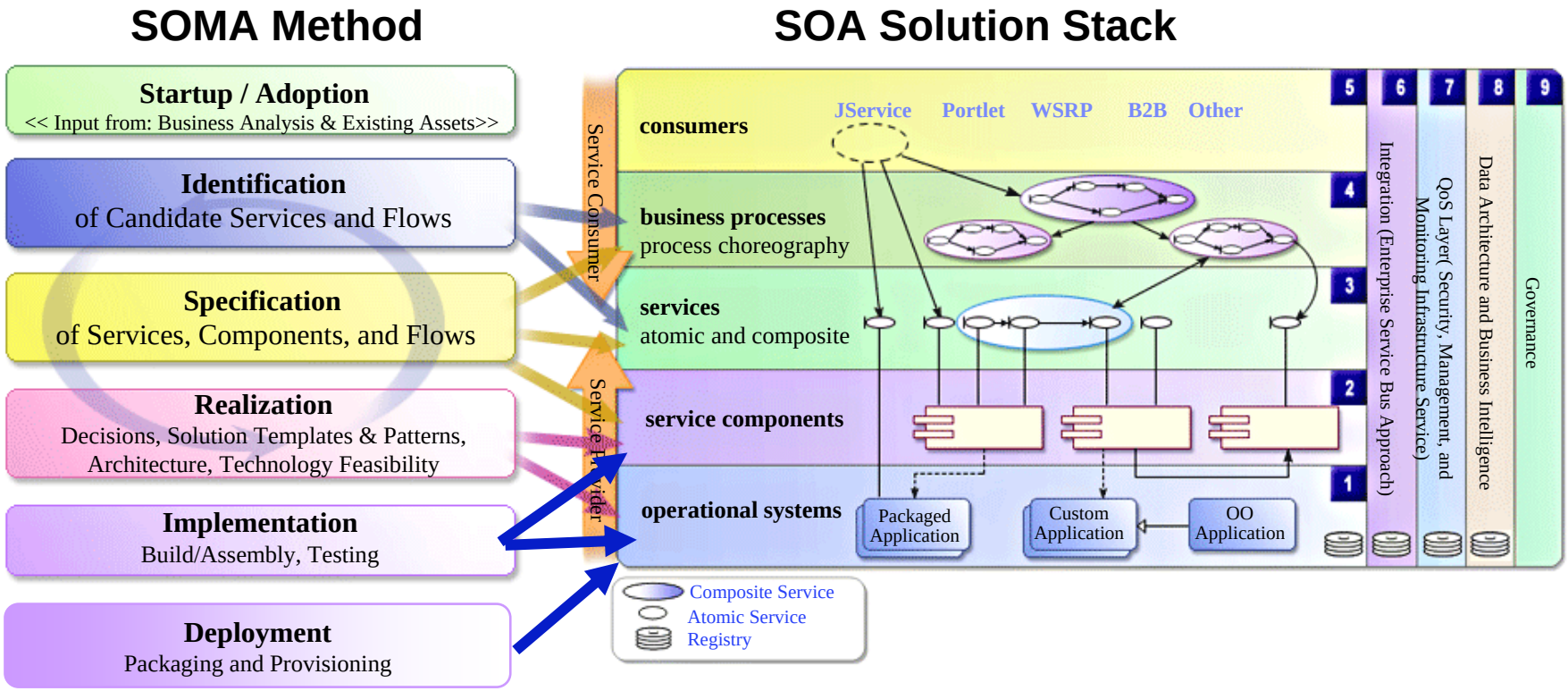


The SOA Solution Stack: Layered solution view

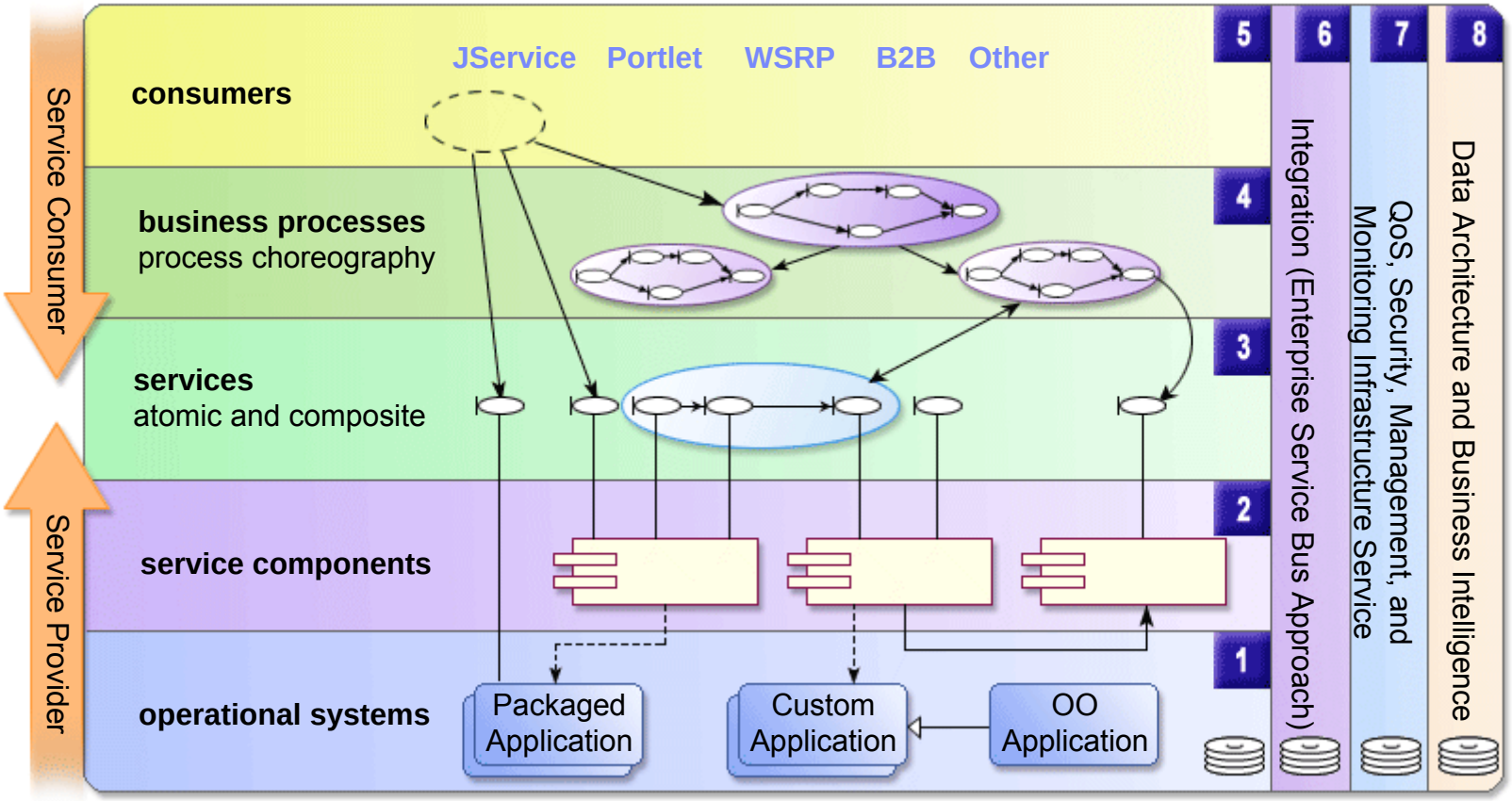


SOMA (Service Oriented Modeling and Architecture) provides SOA Methodology

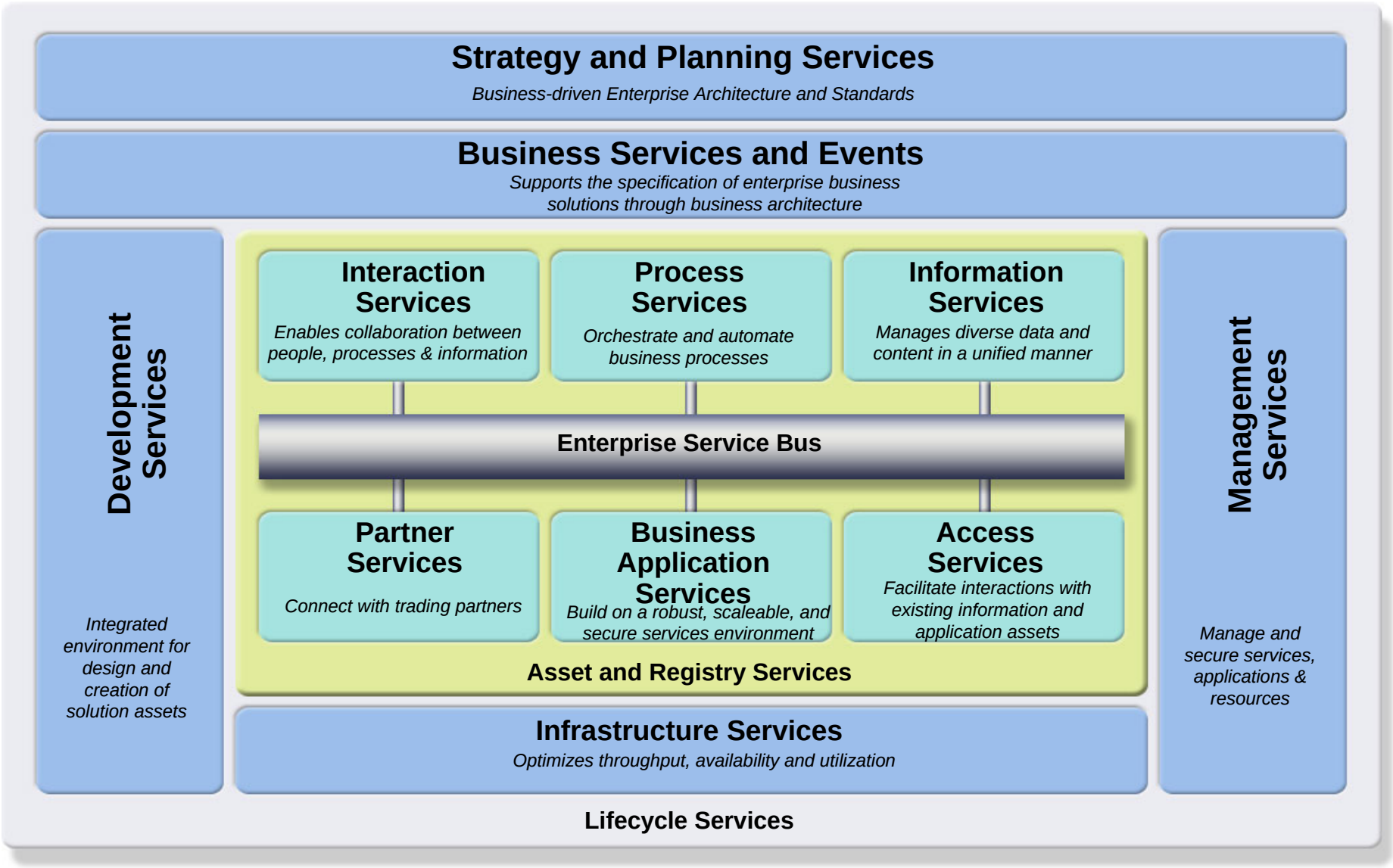
SOMA is about identification, specification, realization, implementation, and deployment of services, components and flows



SOA Layered View (Solution Stack)



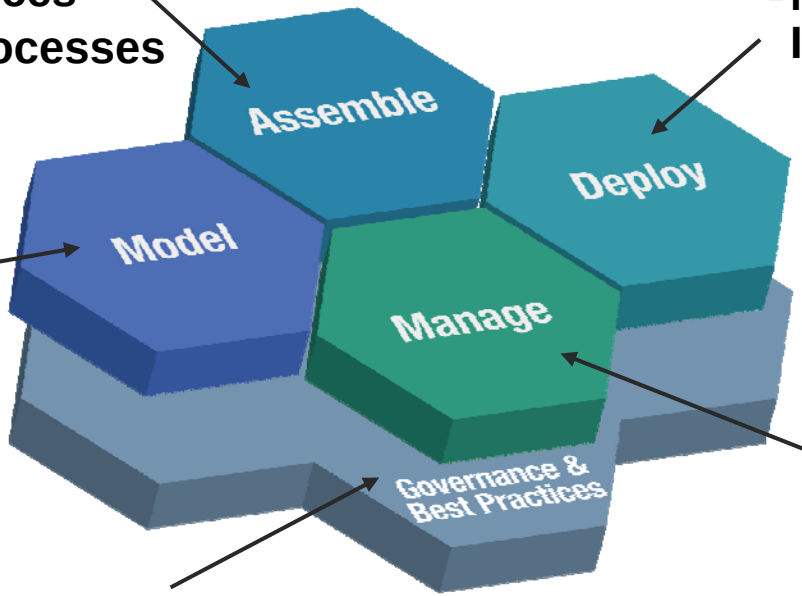
IBM SOA Foundation Reference Model



The SOA Lifecycle (to be addressed in detail in Governance)

- Discover Services
- Construct & Test Services / UIs
- Compose Services
- Orchestrate Processes

- Integrate People
- Integrate Processes
- Manage and integrate Information



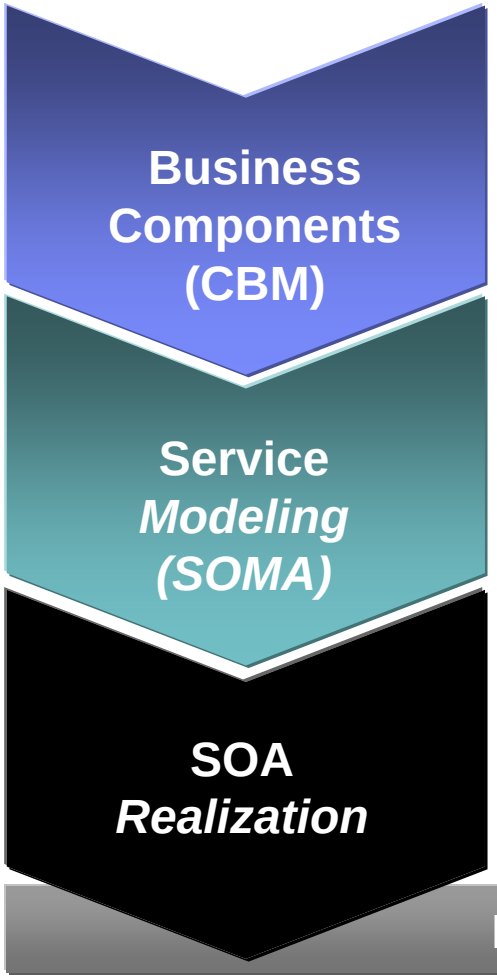
- Gather & Model Requirements
- Model & Simulate
- Model SW Architecture

- Financial Transparency
- Business/IT Alignment
- Process Control

- Manage Applications & Services
- Manage Identity & Compliance
- Monitor Business Metrics

Identification and Specification of Services (SOMA)

Top-Down (Ideal) Approach for SOA Start with Business Design



Step 1: Break down your business into components

- **Decide what is strategically important, and what is just operations in the value chain domains**
- **Analyze the different KPIs attached to these components**
- **Prioritize and scope your transformation projects**

Step 2: Define a Service Model

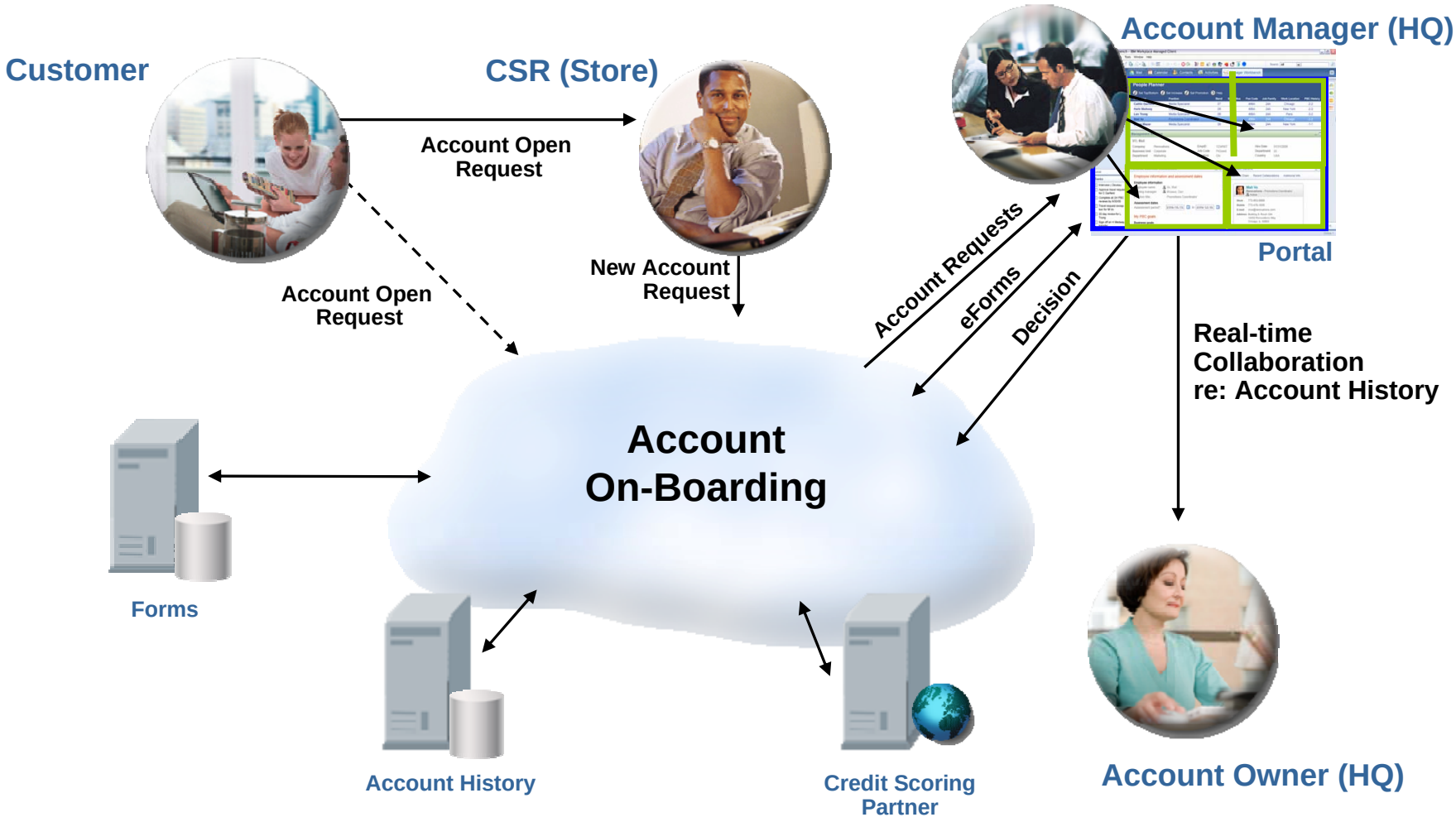
- **Identify your services based on your business components**
- **Specify the services and components accordingly**
- ***Make SOA realization decisions based on architectural decisions***

Step 3: Implement a Service Model

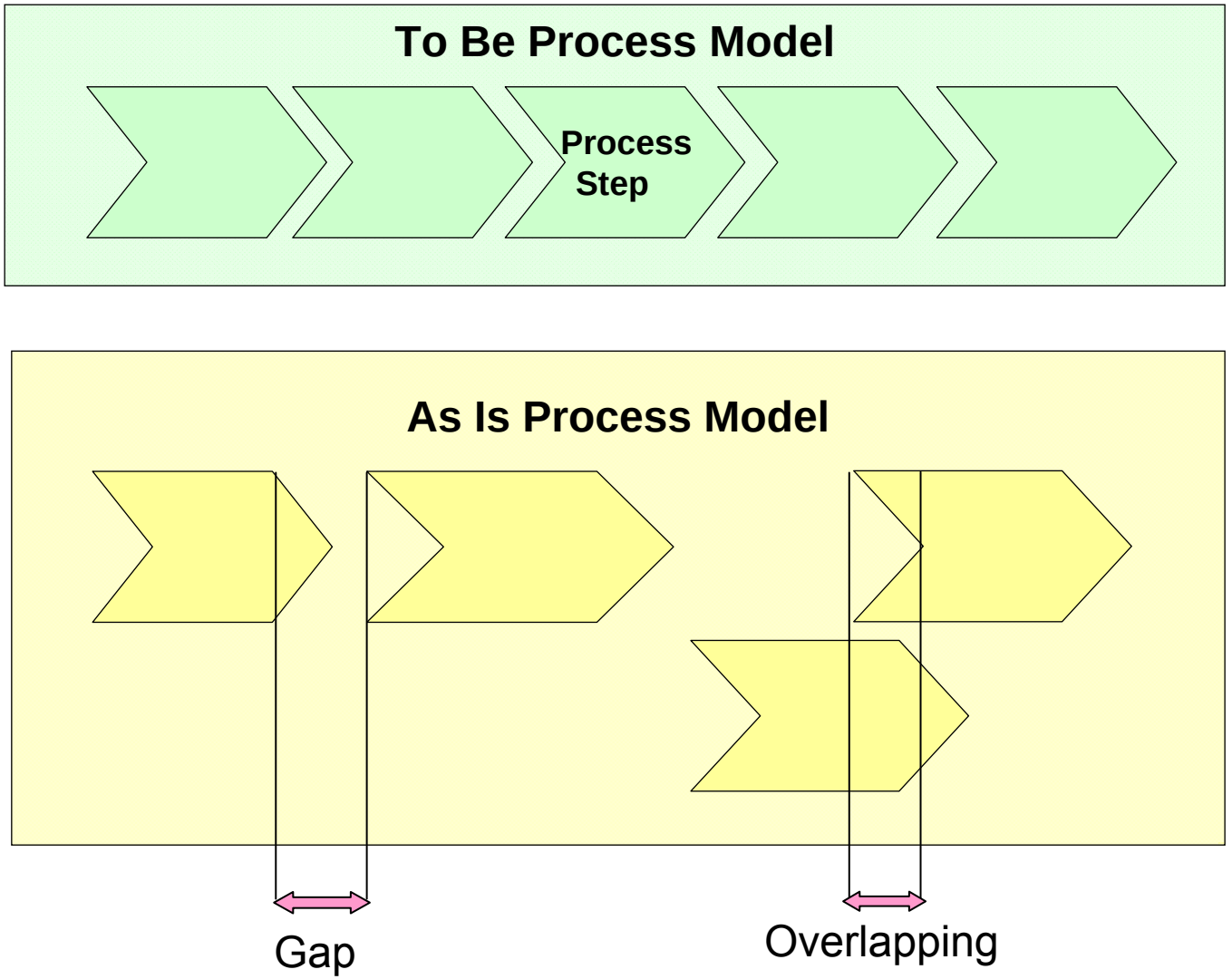
- **Develop a service-oriented architecture to support the Componentized Business**
- **Implement service based scoping policy for projects**
- **Implement appropriate governance mechanism**

Business-Aligned IT Architecture

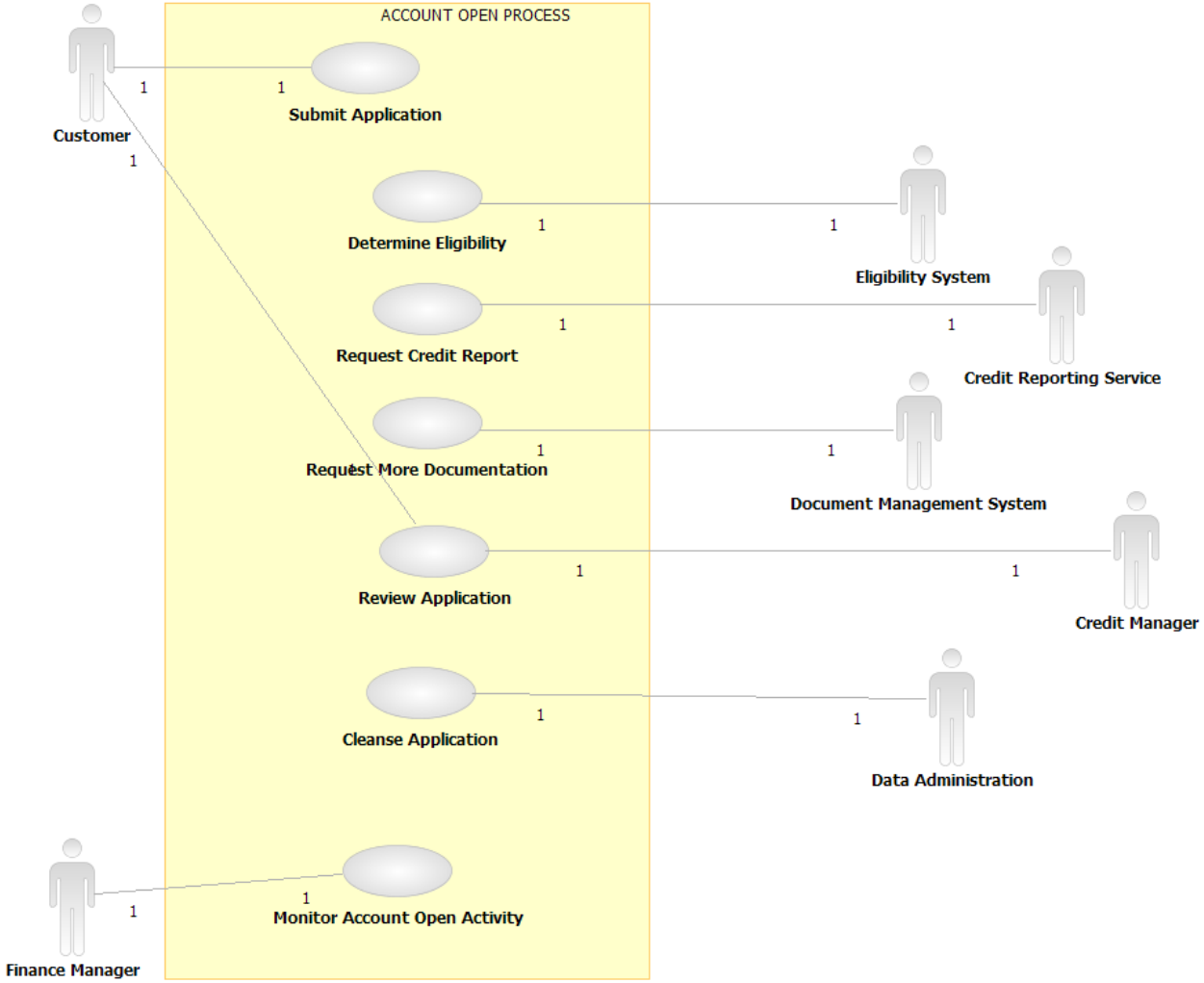
Example: Business Context Diagram for Business Process "Open Account" (Solution Viewpoint)



Business Process Reality and Plans – Streamline Business Process – Derive Requirements



Example: Use Case for JKE's "Open Account"

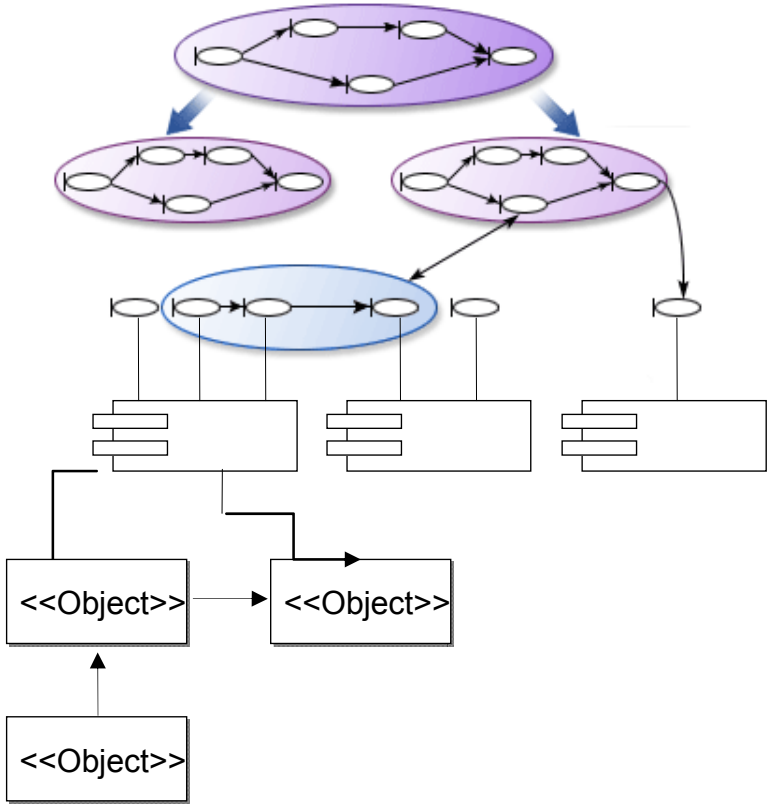


SOA Modeling Constructs

Business Processes
(Flows)

Services
Atomic and Composite

Service Components

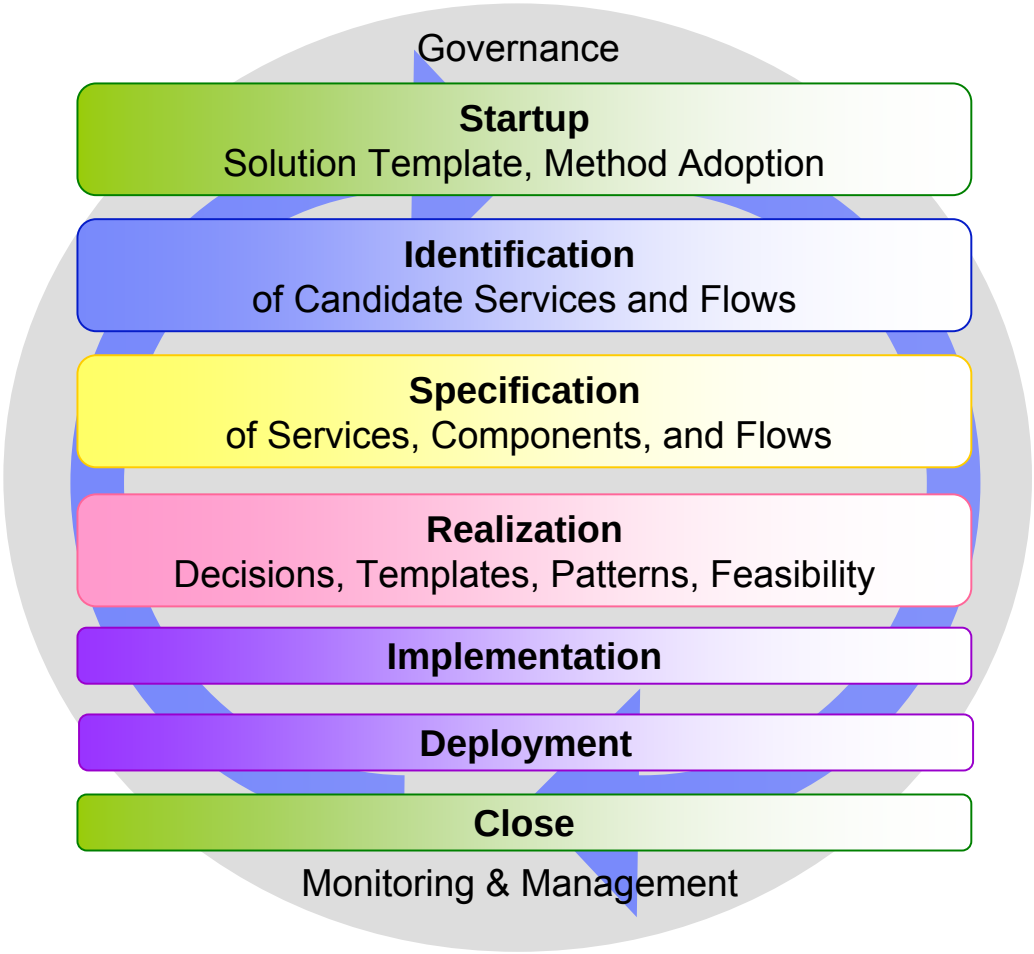


SOMA was created to specifically address modeling of all three constructs.

Introducing SOMA (Service Oriented Modeling and Architecture)

- SOMA is a business-driven modeling and design method
- SOMA provides in-depth guidance on how to move from the business models to the IT models required by SOA
- SOMA adds new service-oriented aspects and techniques in intelligent ways to enable an SOA with services directly traceable to business goals and requirements

At the heart of SOMA is identification, specification, realization and implementation of services, components and flows



- **Design is separated in Identification and Specification**
- **Realization are mainly decisions on how to implement, buy, or use existing assets**
- **Implementation and Deployment as "classical" Software Engineering**

SOMA defines What we do and How we do it

What we do ?

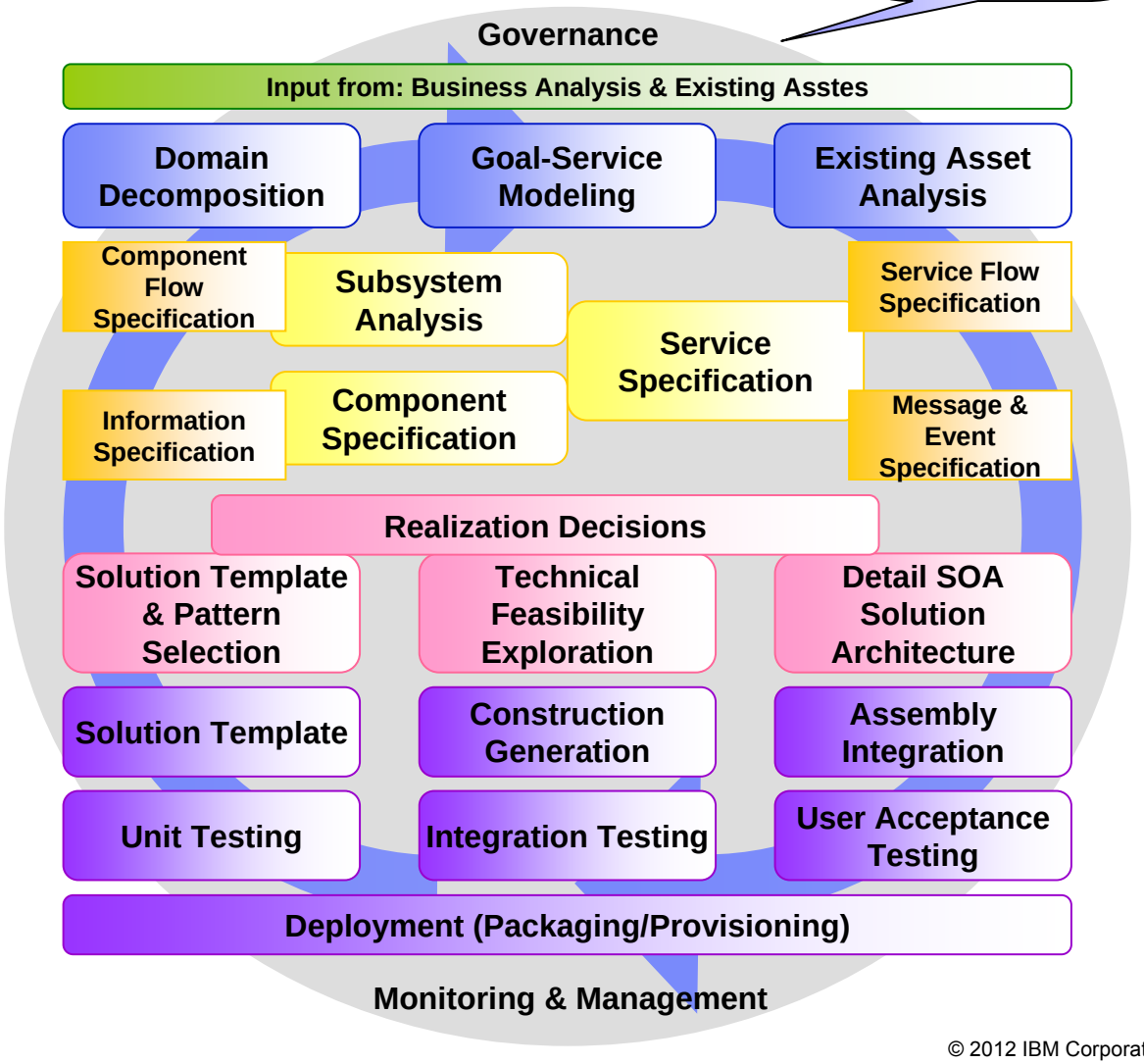
Identification
of candidate services and flows, leverageable existing assets

Specification
of services to be exposed, flows, and components (for realization of functionality)

Realization
captures realization decisions, selects solution templates, details SOA Solution Ref. Arch.

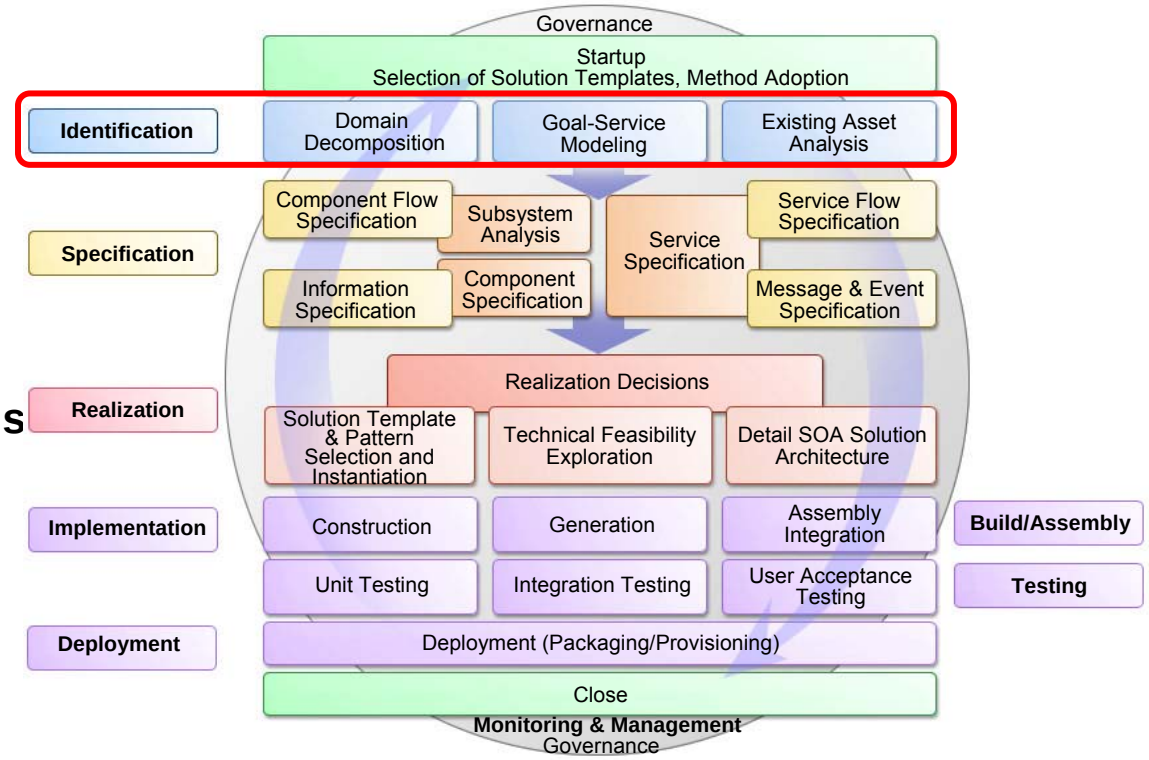
Implementation
incl. construction/ generation, assembly, testing, deployment, monitoring and management

How we do it?



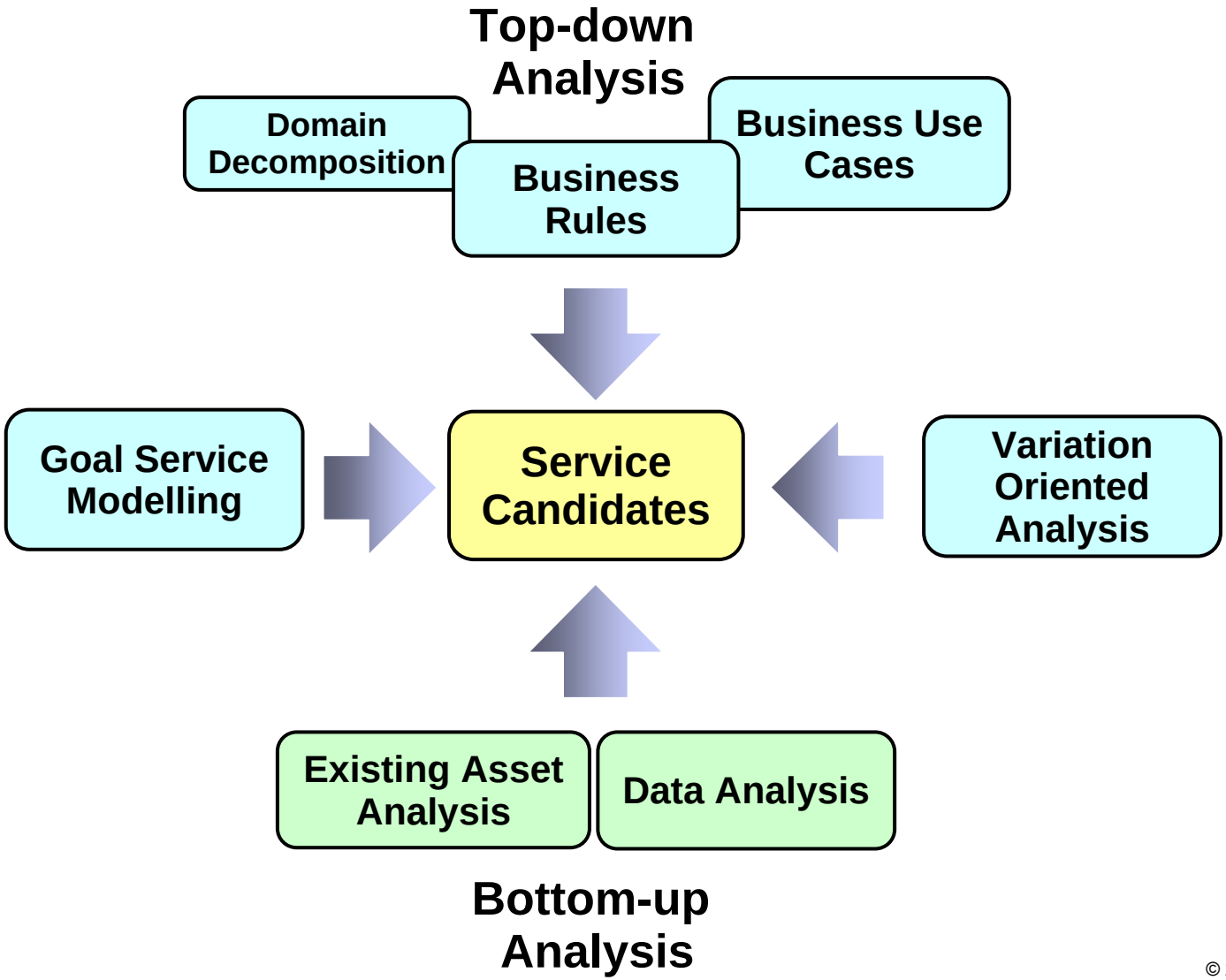
Identifies Services

- **Domain Decomposition (Top-down Analysis)**
  - **Process Decomposition**
  - **Functional Area Analysis**
  - **Information Analysis, Modeling, and Planning**
  - **Rule and Policy Analysis**
  - **Variation-Oriented Analysis**
- **Existing Asset Analysis (Bottom-up Analysis)**
- **Goal-Service Modeling**
- **Additionally, Service Refactoring and Rationalization**
  - **Service Litmus Tests**
  - **Exposure Decisions, including Exposure Scope**

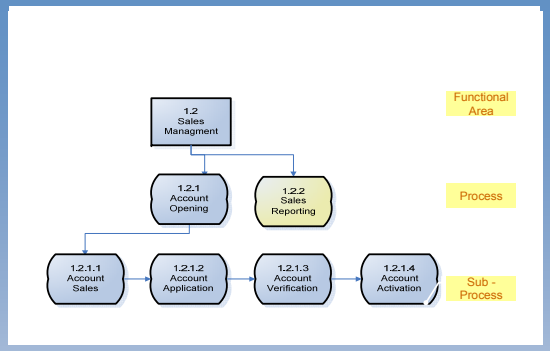


Id Services, Components, and Flows

Service Identification Through 3 main Complimentary Techniques

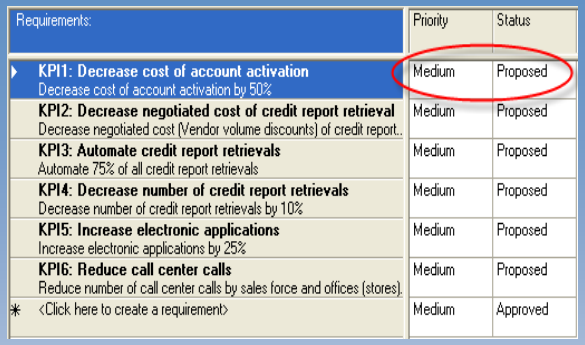


Service Design via SOMA – Service Identification



Domain Decomposition

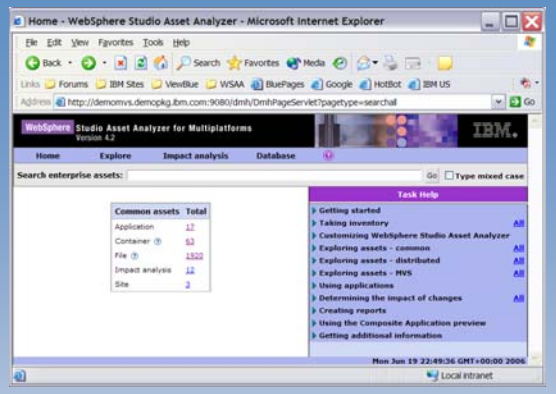
- **Techniques:**
  - **Process Modeling Tools**
  - **Design of KPIs/Metrics**
- **Services Identified**
  - **Open Account**
  - **Account Activation**
  - **Account Verification**



Requirements:	Priority	Status
KPI1: Decrease cost of account activation Decrease cost of account activation by 50%	Medium	Proposed
KPI2: Decrease negotiated cost of credit report retrieval Decrease negotiated cost (Vendor volume discounts) of credit report..	Medium	Proposed
KPI3: Automate credit report retrievals Automate 75% of all credit report retrievals	Medium	Proposed
KPI4: Decrease number of credit report retrievals Decrease number of credit report retrievals by 10%	Medium	Proposed
KPI5: Increase electronic applications Increase electronic applications by 25%	Medium	Proposed
KPI6: Reduce call center calls Reduce number of call center calls by sales force and offices (stores).	Medium	Proposed
<Click here to create a requirement>	Medium	Approved

Goal Service Modeling

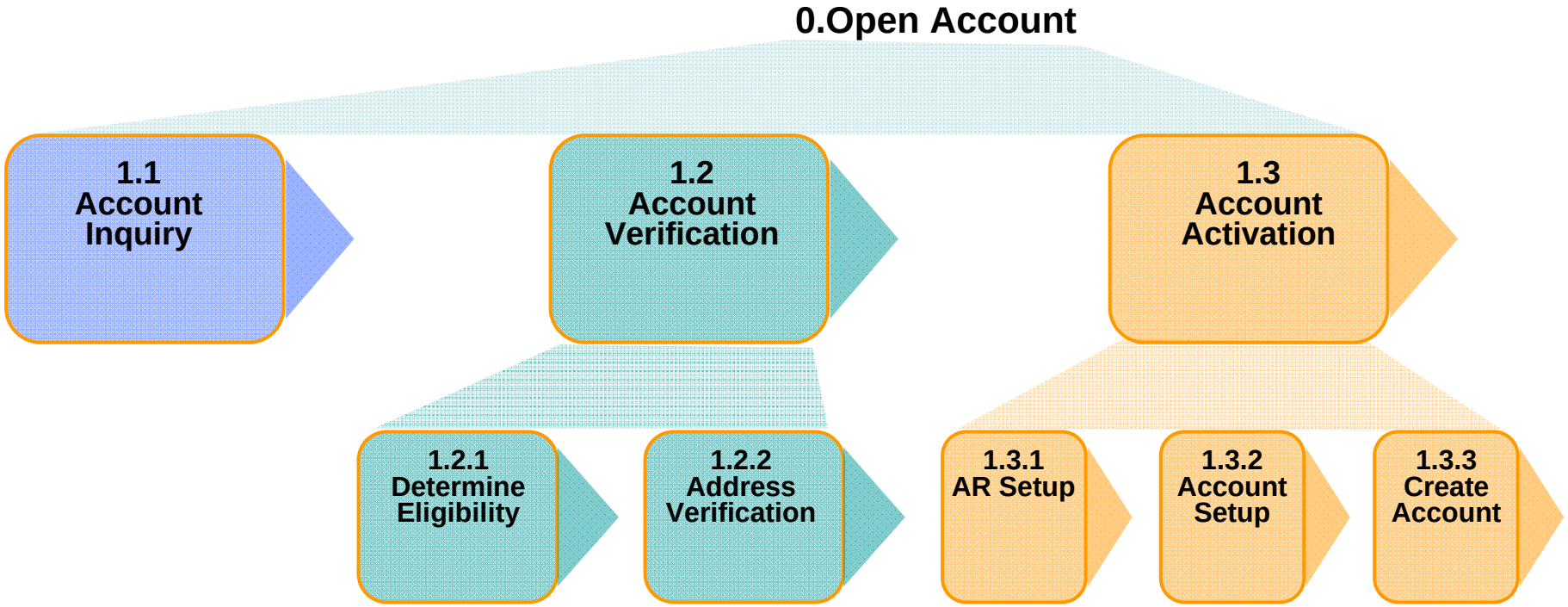
- **Techniques**
  - **Requirements Planning Tools**
  - **Design of KPIs/Metrics**
- **Services Identified**
  - **Determine Applicant Eligibility**
  - **Address Verification**



Existing Asset Analysis

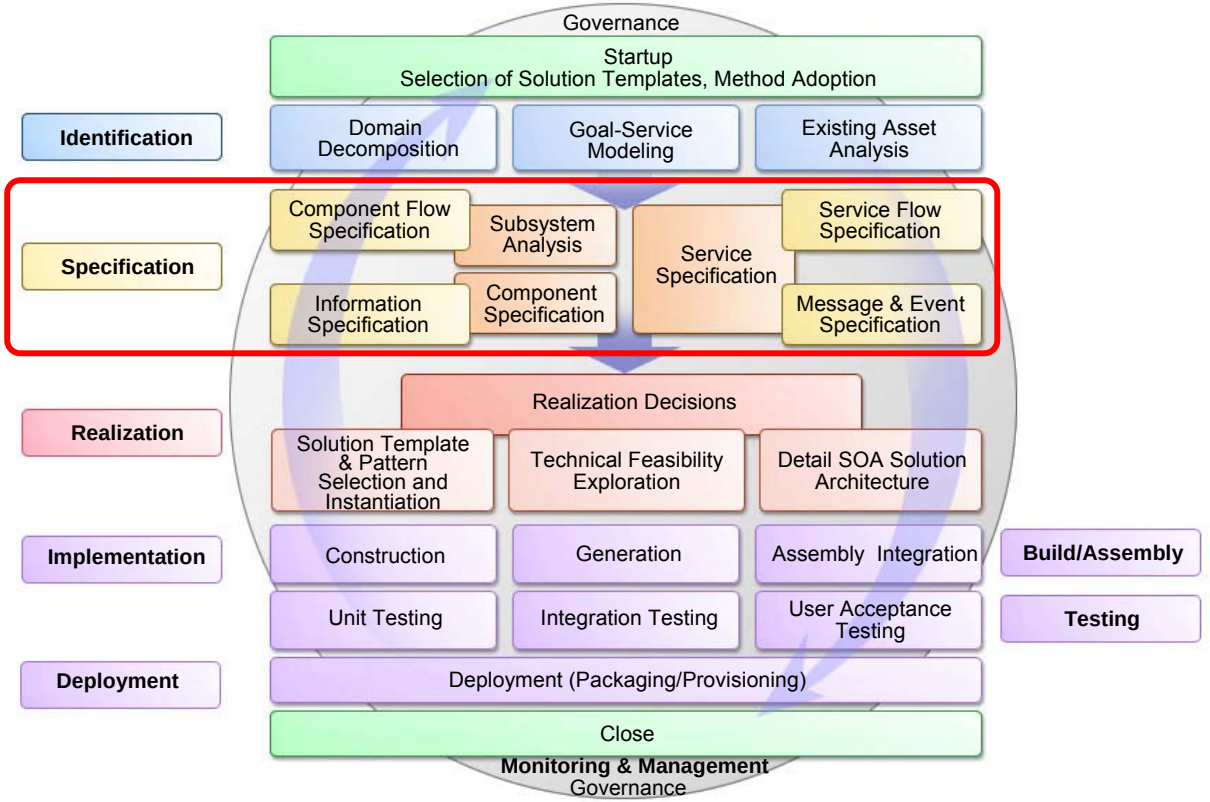
- **Techniques**
  - **Asset Analysis Tools**
  - **Interviews/Documentation**
- **Services Identified**
  - **Account Inquiry (CICS 2.2)**
  - **AR Setup (CICS 2.2)**
  - **Account Setup (CICS 3.1)**
  - **Create Account (SAP)**

Example: Domain Decomposition – Business Process Modeling for JKE's "Open Account"



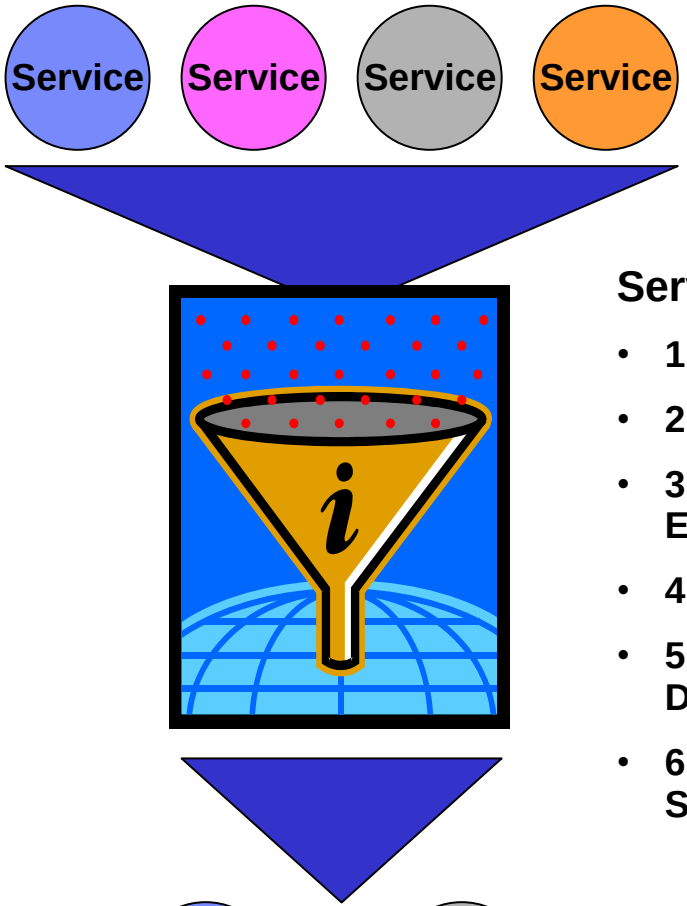
SOMA Specification uses comprehensive techniques to specify Services, Flows, and Service Components that Realize Services

- **Information Specification**
  - **Data Model, Message Model, Business Glossary**
- **Existing Asset Analysis – Fine Grained**
  - **Determine the technical viability of existing applications and approaches to realize services**
- **Service Specification**
  - **Elaborates the Service Model, for example, service dependencies, service composition and flow, rules and policies, event specification, service operation, service message specification, QoS requirements, design decisions, and so on**
- **Subsystem Analysis**
  - **Partitions subsystems into service components that will be responsible for service realization**
- **Component Specification**
  - **Details component modeling, flow, information architecture, messages**



Service Litmus Tests Are Gating Criteria Used to Determine If a Candidate Service Should Be Exposed

Candidate Services

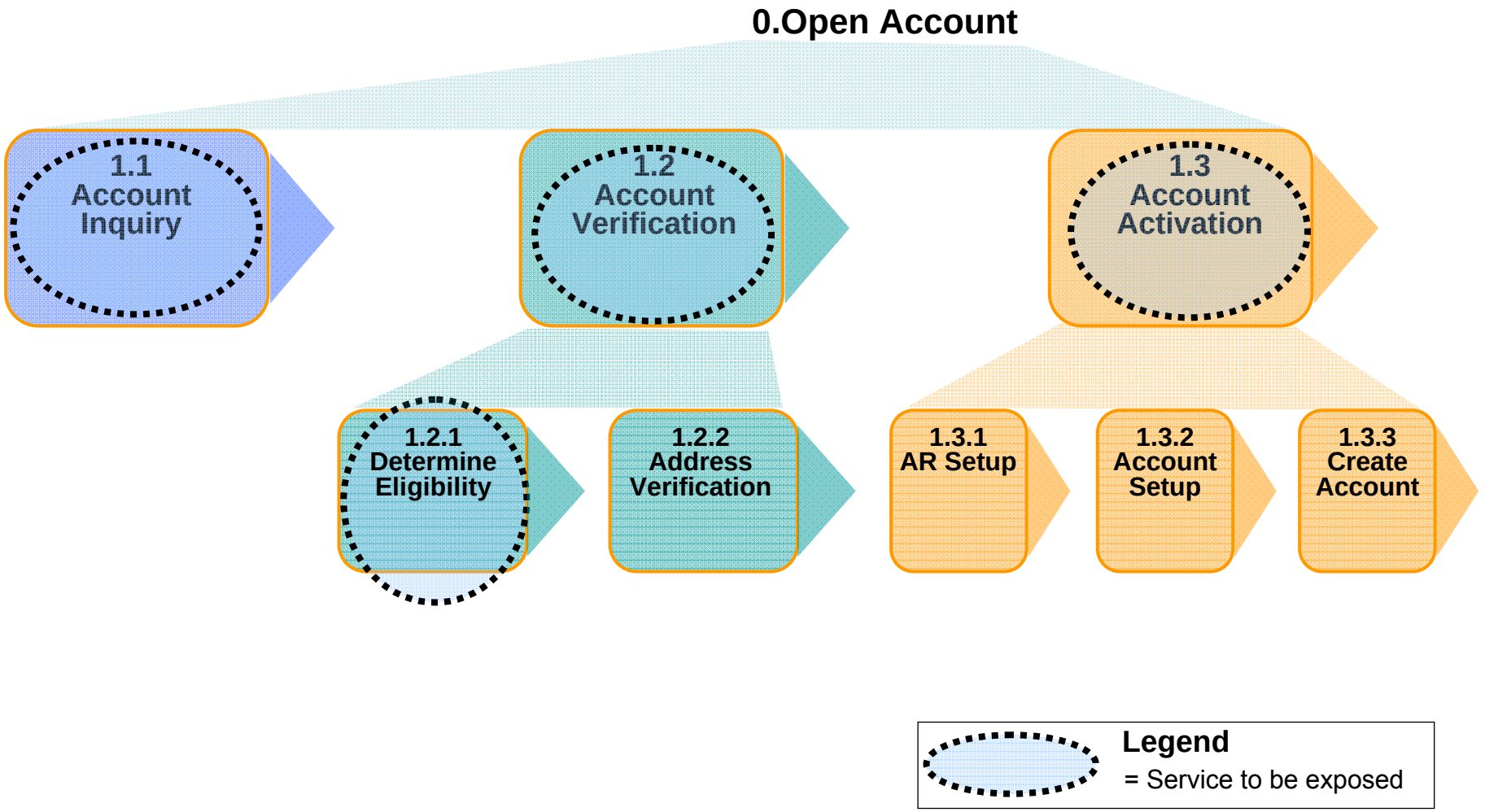


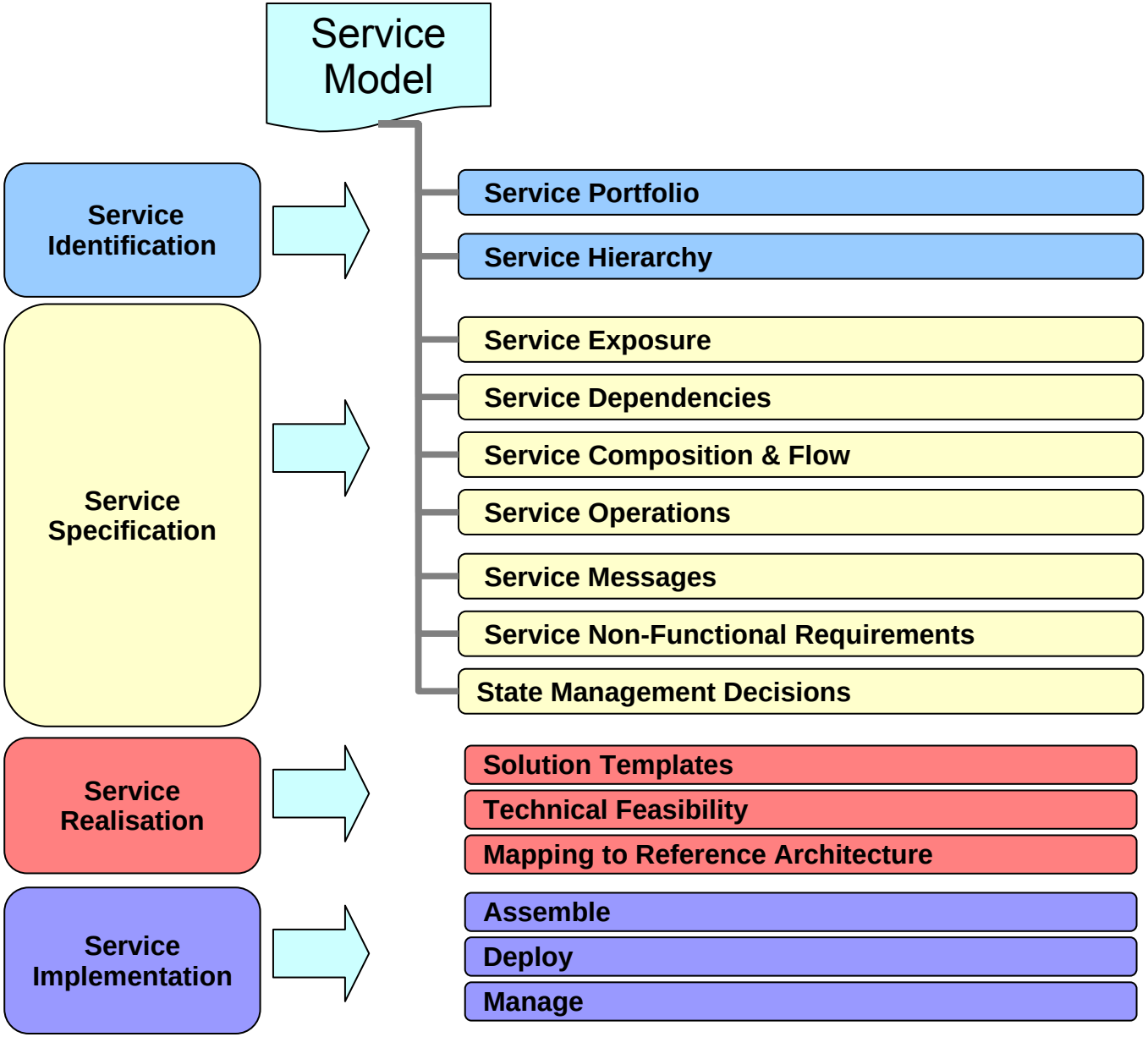
Exposed Services

Service Litmus Tests

- 1. Business Alignment
- 2. Composability
- 3. Consolidation (Redundancy Elimination)
- 4. Technical Feasibility
- 5. [Externalized Service Description]
- 6. Project Defined/Customer Specific SLTs

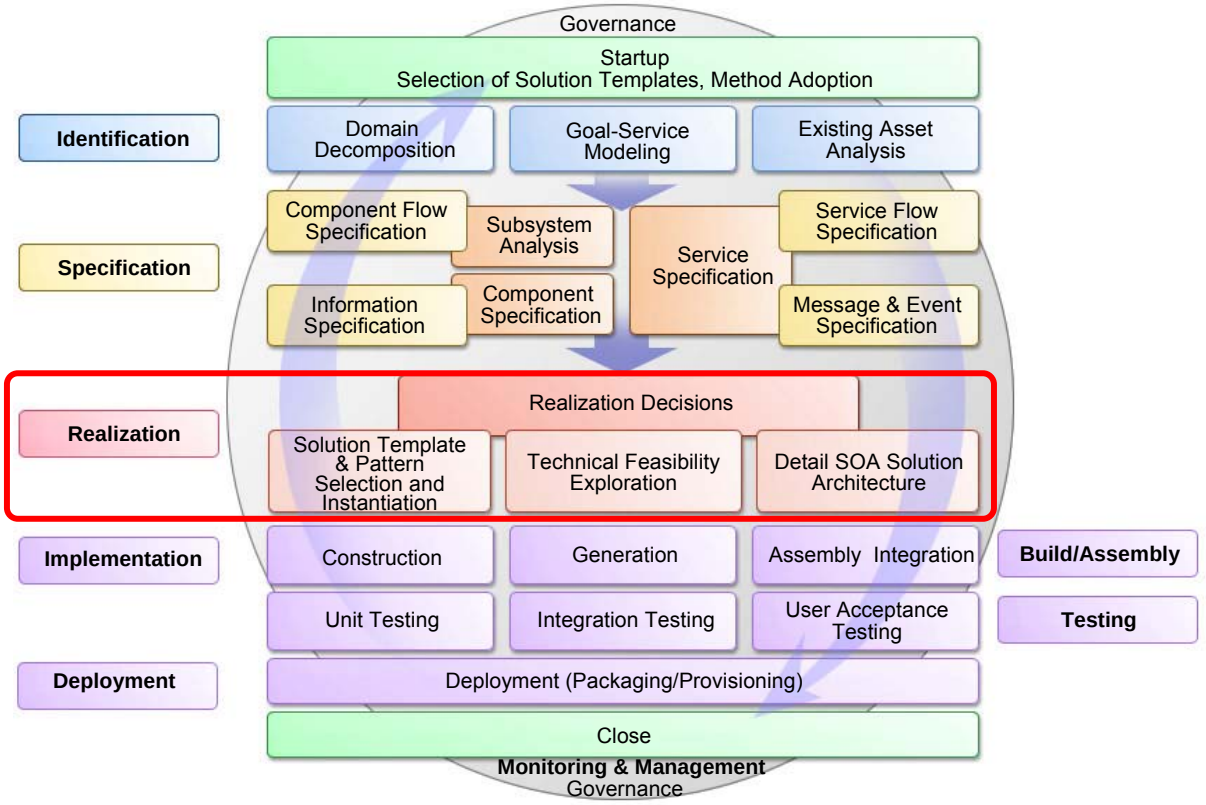
Example: JK Enterprises Service Exposure Decisions





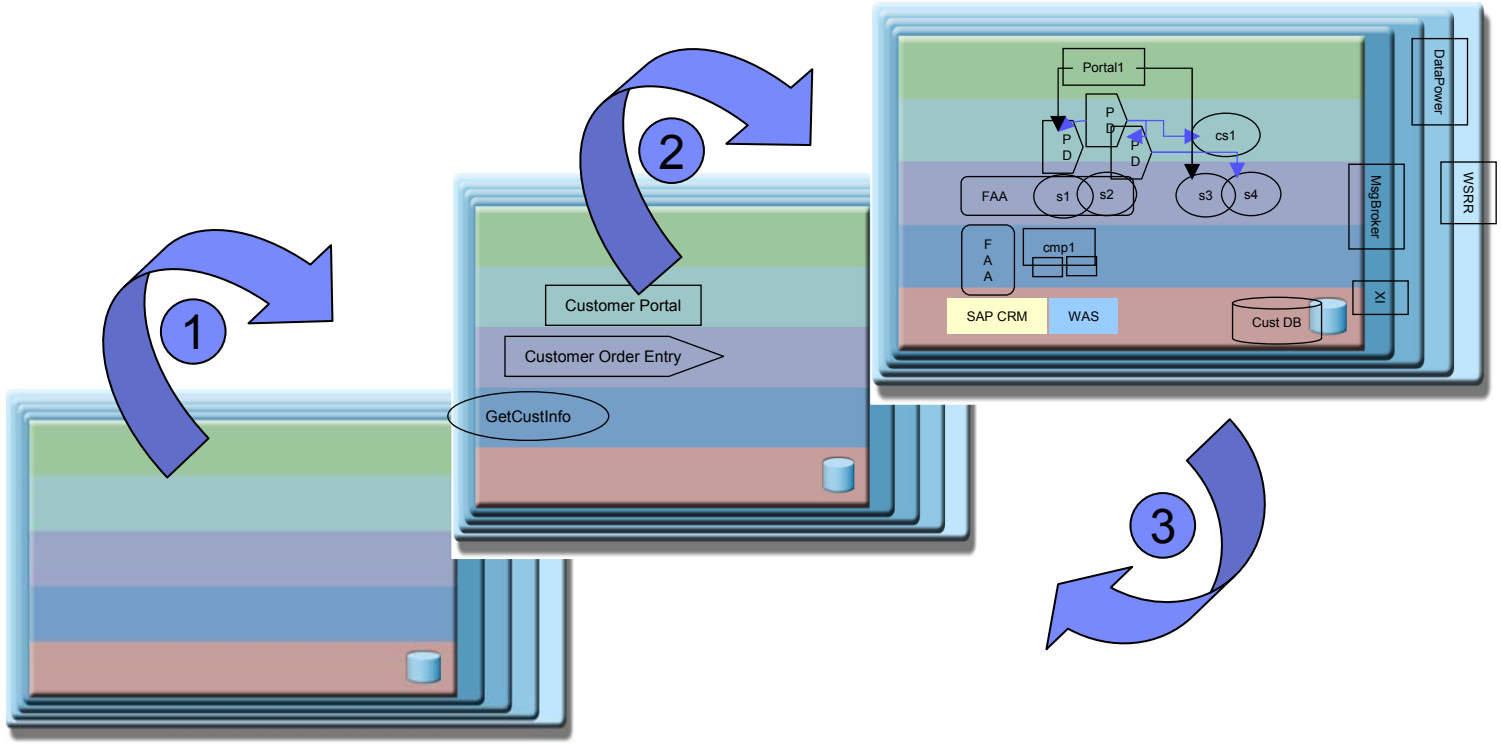
SOMA Realization (Includes SOA Solution Stack Instantiation)

- **Select and instantiate Solution Templates and Patterns**
- **Technical Feasibility Exploration**
  - **Examine approaches to handle client requirements**
  - **Examine legacy application specific considerations**
- **Detail SOA Solution Stack**
- **Realization Decisions**
  - **Consider alternatives**
  - **Select the alternative**
  - **Provide justification**



Iterative SOA Solution Design Process

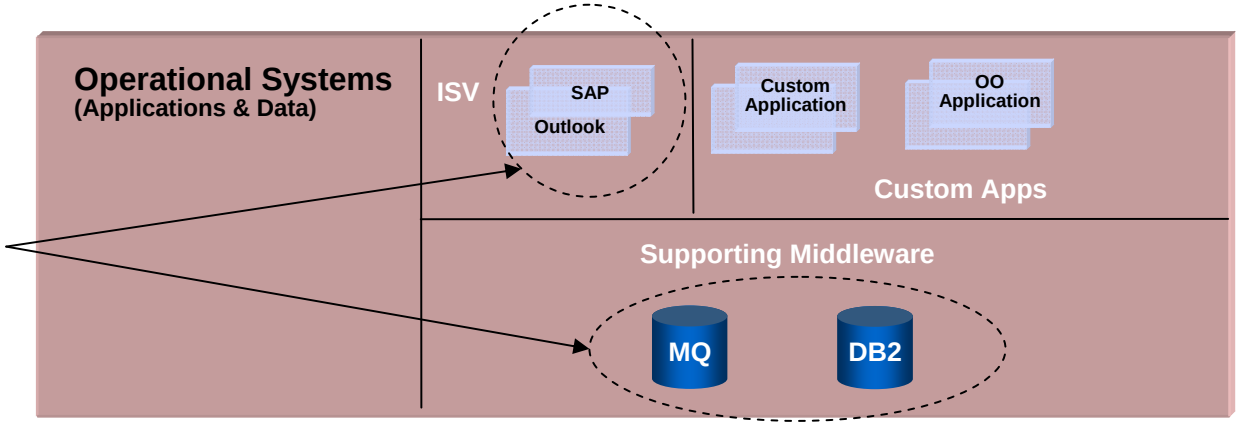
As SOMA is applied during an engagement, we incrementally populate an architectural overview ("dashboard view") of the SOA Solution



SOA Layered View Details

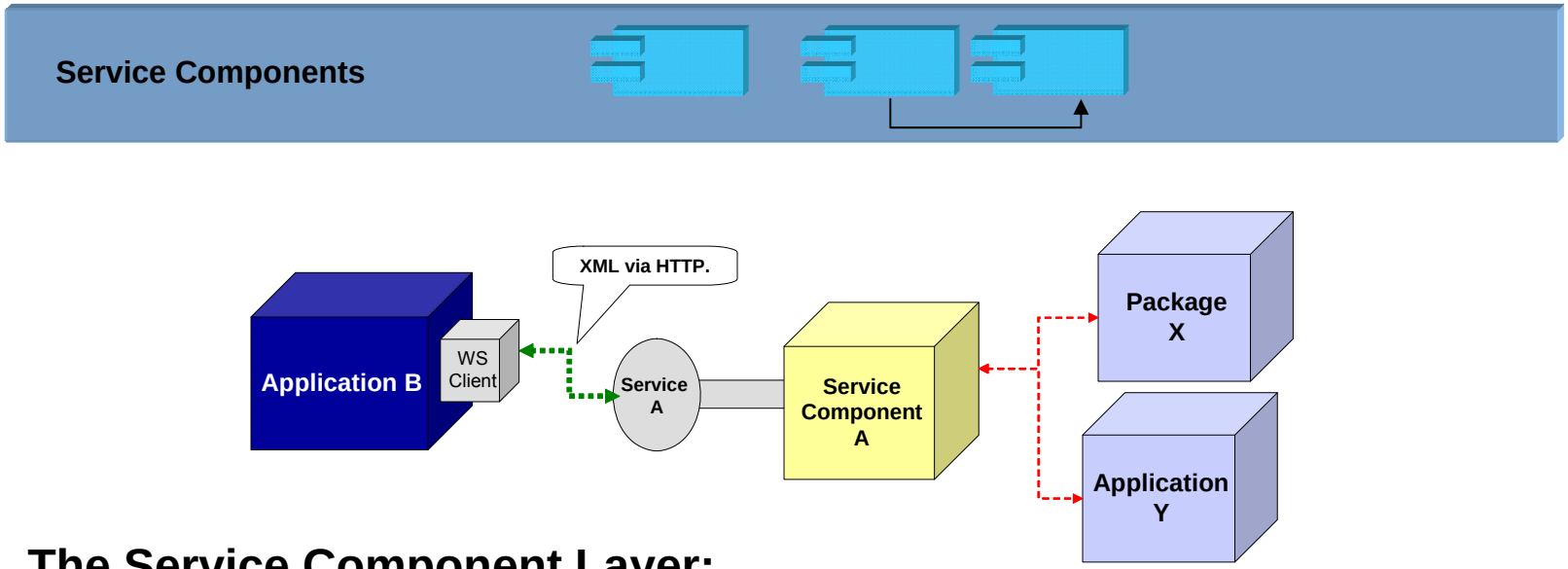
Layer 1: Operational Systems (Leverage Existing Investment)

Examples for illustration: specifics are not in the scope of the reference model.



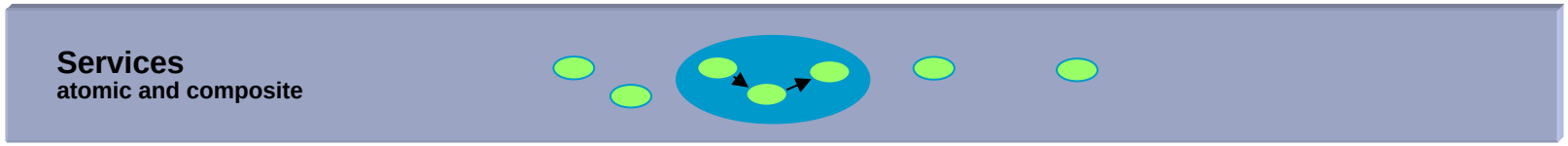
- **Recognizes the value of existing IT investment**
  - **Use of existing "legacy" applications (e.g. COBOL application) and / or packages (e.g. SAP)**
- **Some SOA Related Activities:**
  - **Asset Inventory**
  - **Refactor existing applications to unlock business value**

Layer 2: Service Components



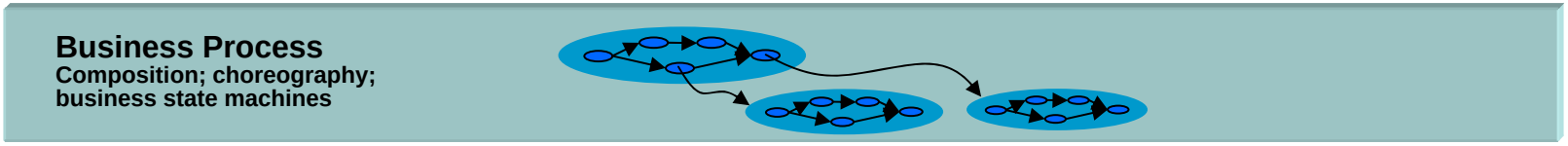
- **The Service Component Layer:**
  - **Enables IT flexibility by strengthening the decoupling in the system. Decoupling is achieved by hiding volatile implementation details from consumers.**
  - **Often employs container based technologies like EJBs**
- **Each Service Component:**
  - **Provides an enforcement point for service realization**
  - **Offers a facade behind which IT is free to do what they want/need to do**

Layer 3: Services (Decouple Business and IT)



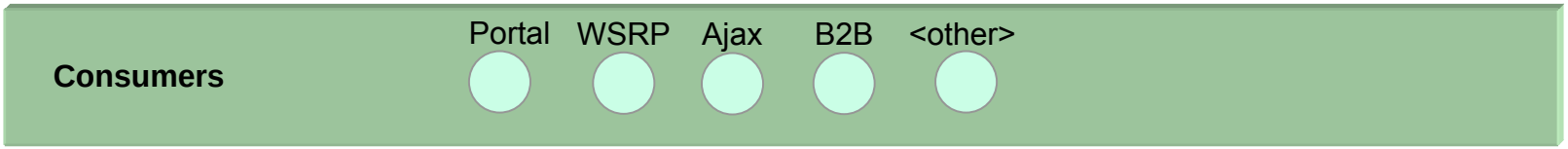
- **The Services Layer forms the basis for the decoupling of Business and IT.**
  - **Captures the functional contract (incl. QoS – Quality of Service) for each standalone *business* function or each task in a business process**
- **The assumption is that (within an SOA) IT responsibility is to realize/manage service implementations that faithfully conform to the set of services in the service model.**
- **This layer contains all the *exposed services* in the SOA**
- **Each service is a contract between the consumer(s) and the provider(s)**

Layer 4: Business Processes (Business process alignment of IT)



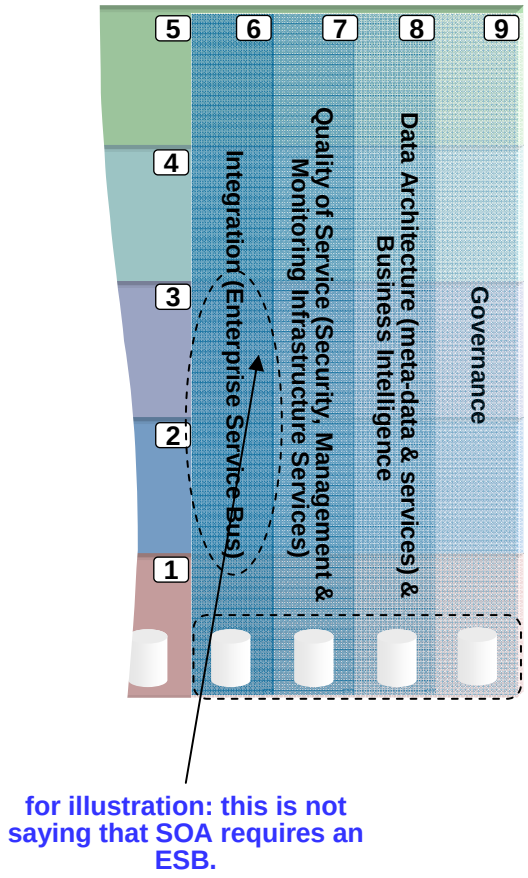
- This layer contains operational IT artifacts that implement business processes as a choreography of services
- The set of services that are composed is restricted to those services that are defined in Layer 3
- The choice of technology depends on a set of realization decisions that must be made when establishing a physical Reference Model for a given SOA

Layer 5: The Consumer Layer (Channel independent access to business processes)



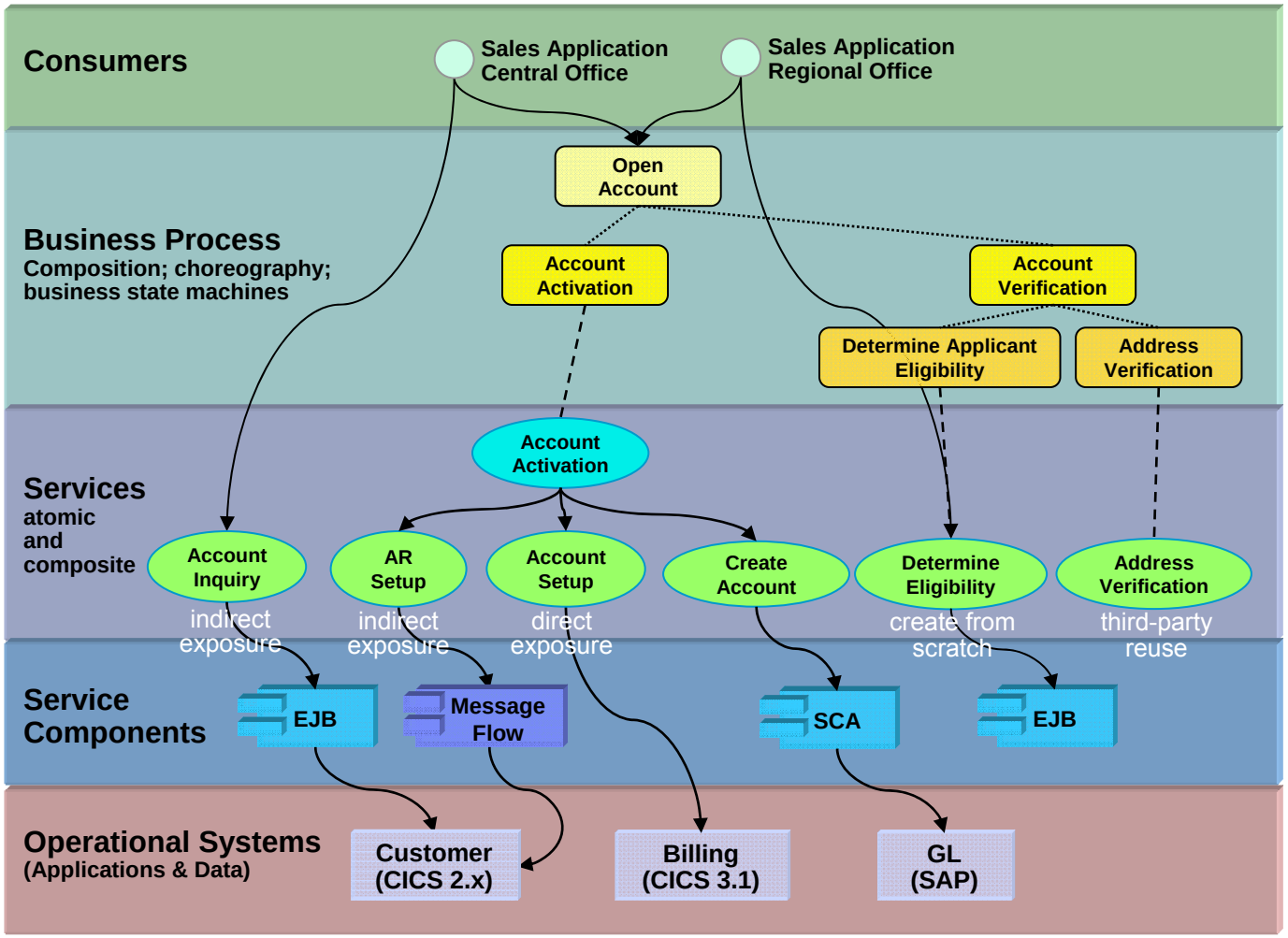
- **This layer exists to recognize that the technology chosen to expose Business Processes/Services must permit access from a wide set of interaction *channels*.**
- **It is important to populate this layer with the set of *channels* types that are required in a solution.**

Cross-cutting concerns/capabilities

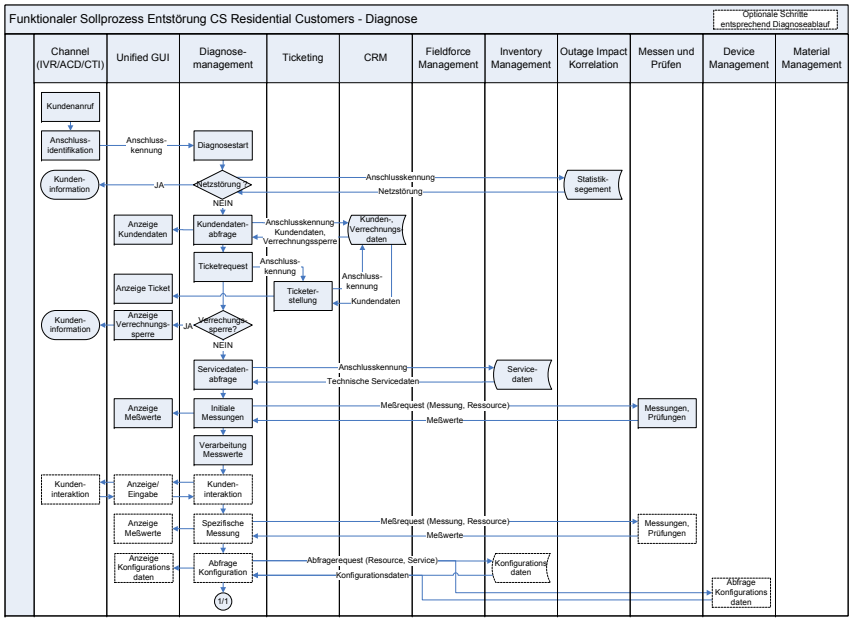


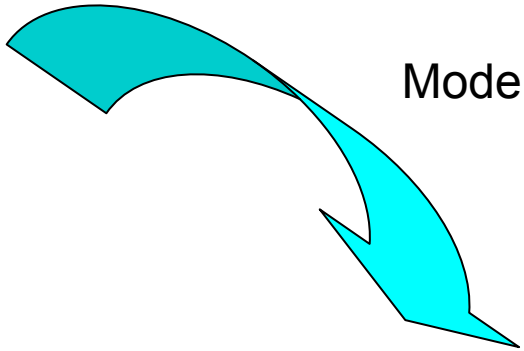
- Several concerns are not restricted to a single layer in the Reference Model, these concerns are captured in ‘Layers’ 6-9
- These are not really layers but treating them as such gives us the ability focus discussions/decisions, for example “What is found where Governance intersects Services? i.e. what are the Governance concerns specific to Services?”
- Clearly there is interaction among these ‘layers’ also. For example, it is likely that most data architectures will be subject to governance

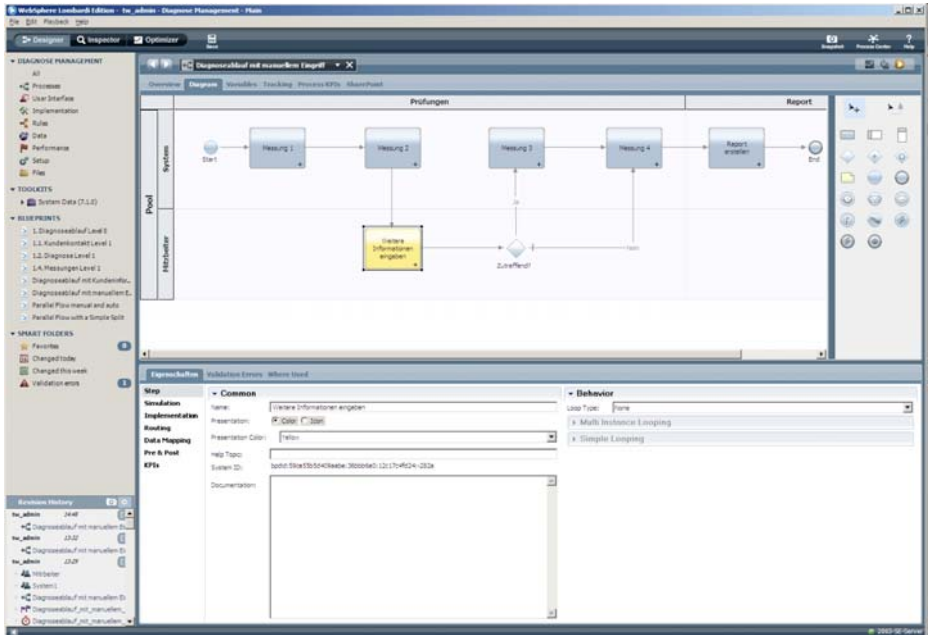
Example JK Enterprise – a virtual company with an „Open Account Process“



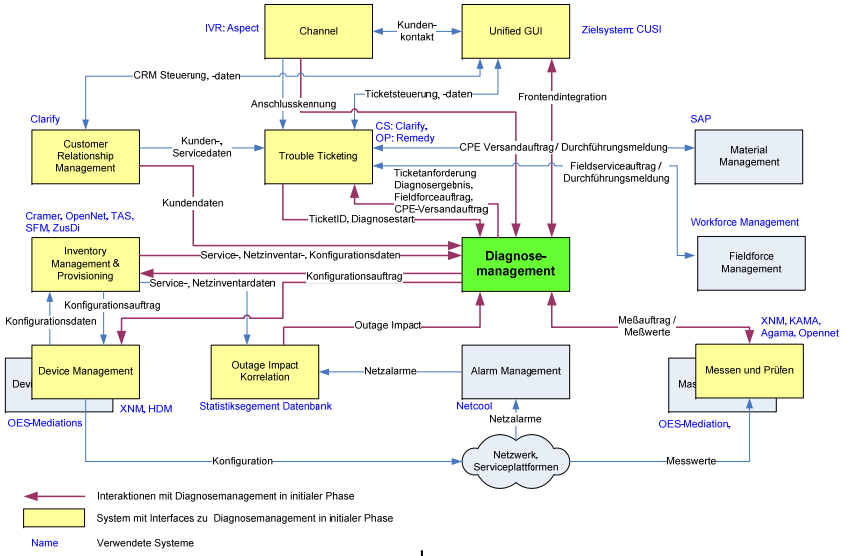
Designing BPM / SOA Application: Process Modeling



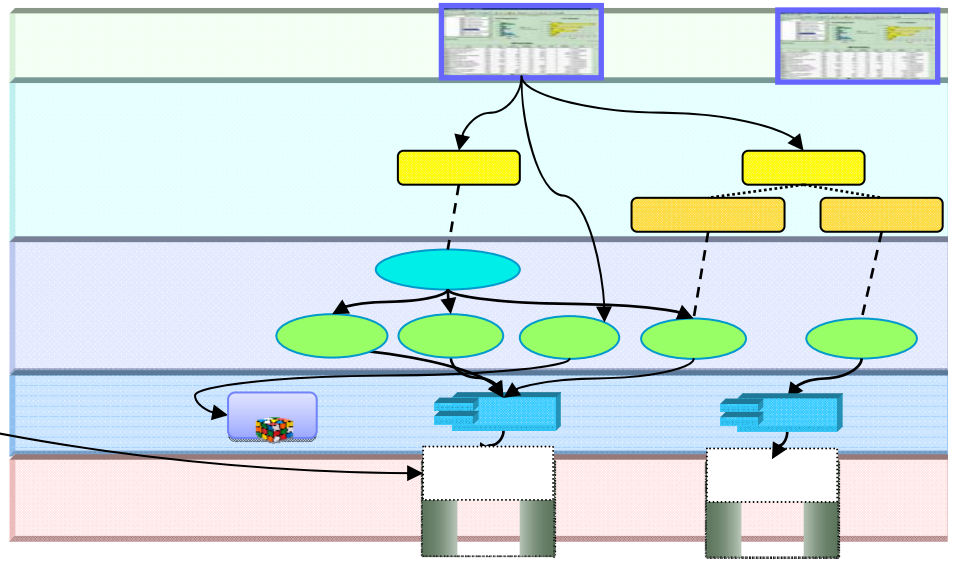




Designing BPM / SOA Application: Layered View



Architecture

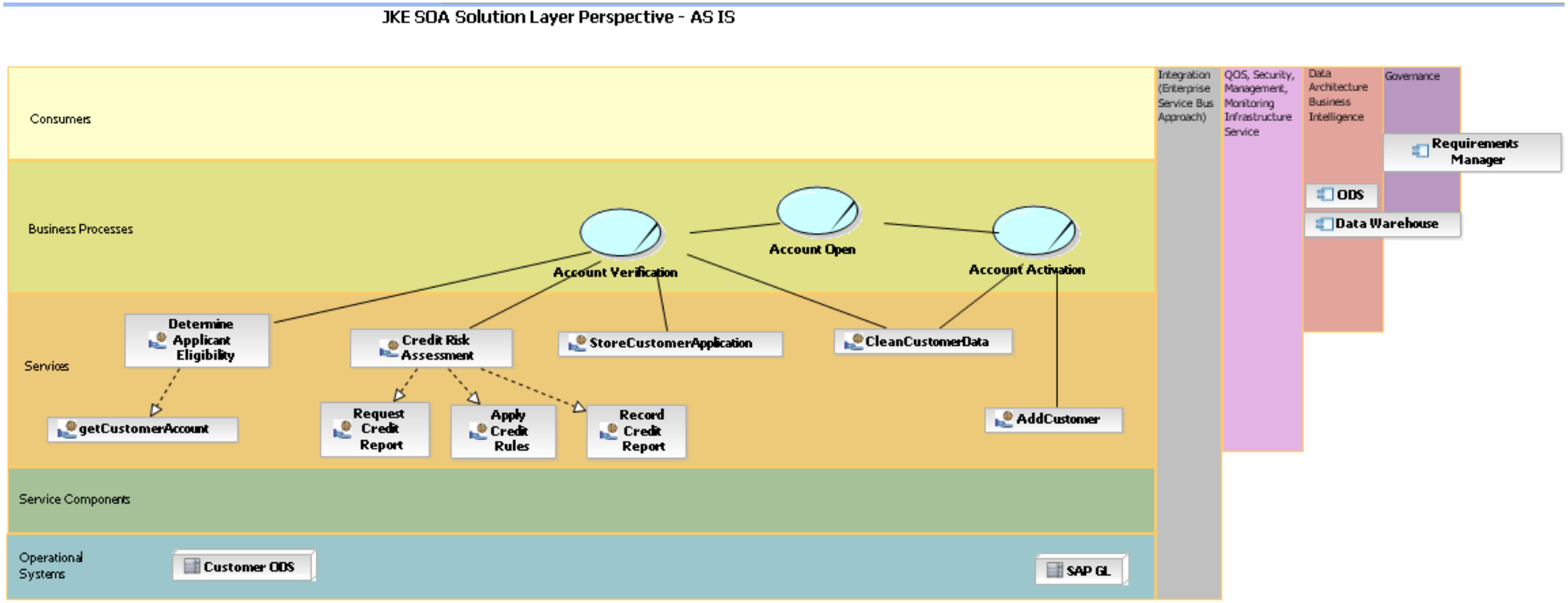


Integration of “Legacy”

Home Work: Exercise Layered View

- Usually a diagram (or set) which is used as a basis for discussion and explanation.
- Assume you will create many iterations of this document.
- Should contain processes, services, components, and operational systems

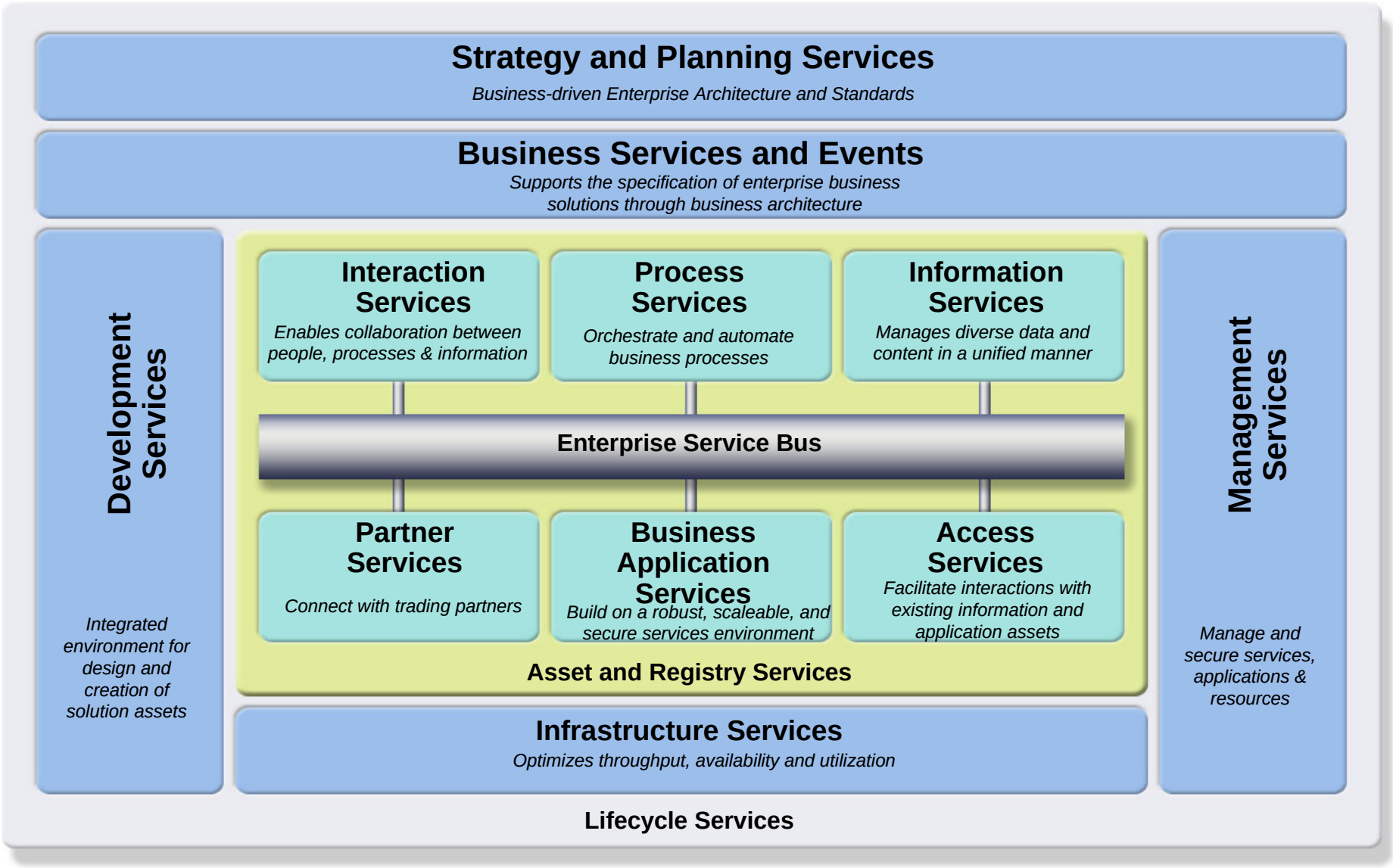
Exercise – SOA Solution Layer Perspective – Add Missing Components



- Among the missing artifacts from this diagram, the Service Components (service realization)
- Also missing are To-Be supporting operational systems

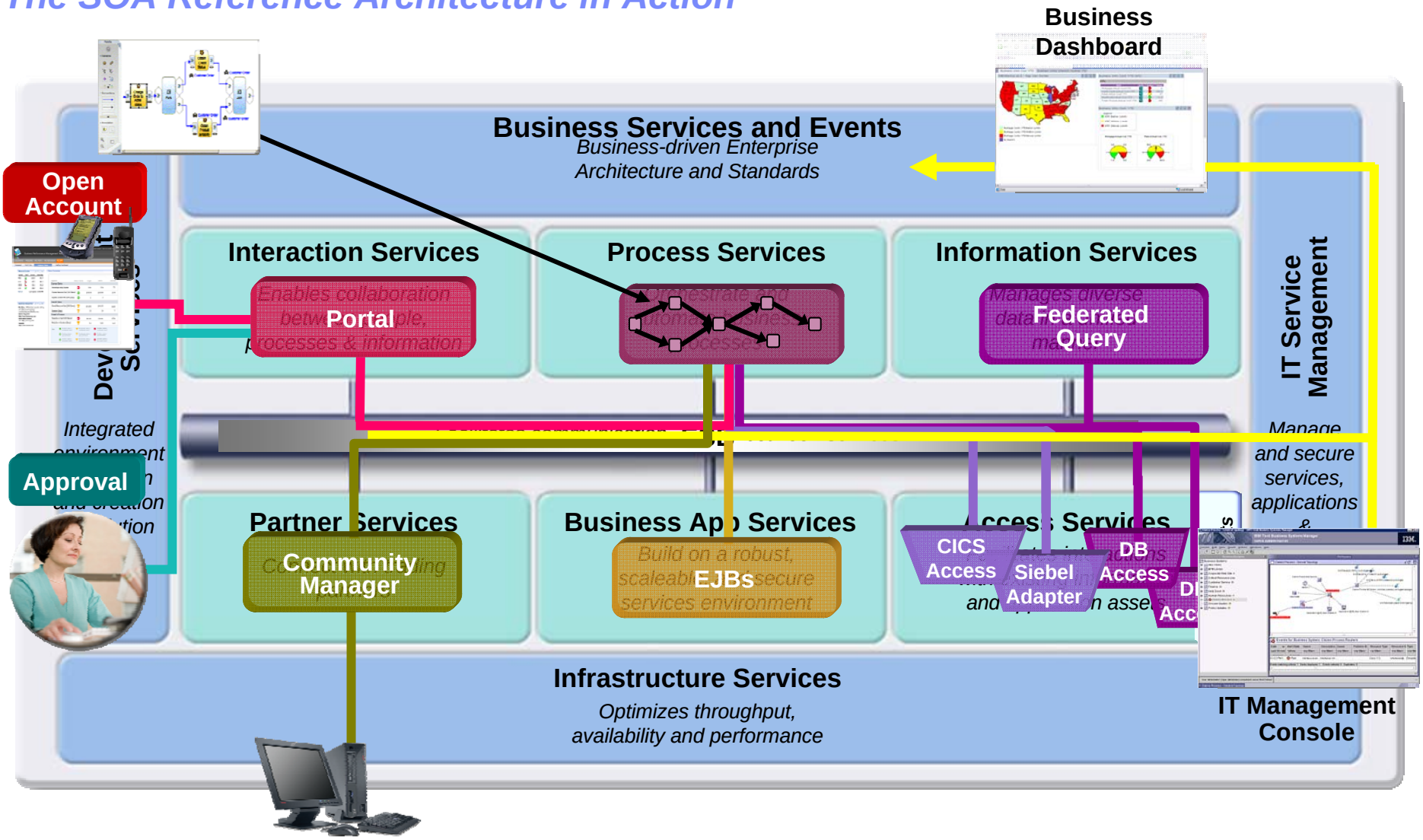
SOA Reference Model

IBM SOA Foundation Reference Model



Separation of Concerns: Example "Open Account" Process

The SOA Reference Architecture in Action



ESB (Enterprise Service Bus)

ESB (Enterprise Service Bus) – Definition and Purpose

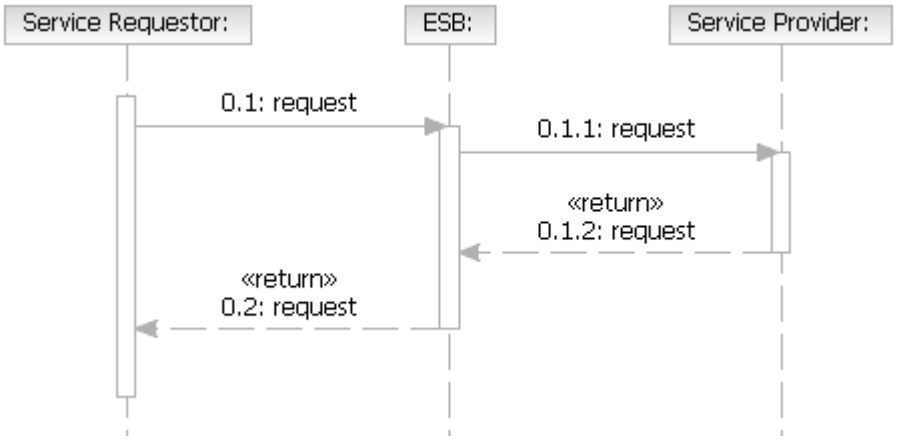
- An Enterprise Service Bus (ESB) is an architectural pattern defining a flexible connectivity infrastructure for integrating applications and services.

- The architecture pattern is a guiding principle to enable the integration and federation of multiple service bus instantiations.

- An ESB performs:
  - Routing messages between services
  - Converting transport protocols between requestor and service – managing multiple protocols
  - Transforming message content between requestor and service
  - Handling business events from disparate sources

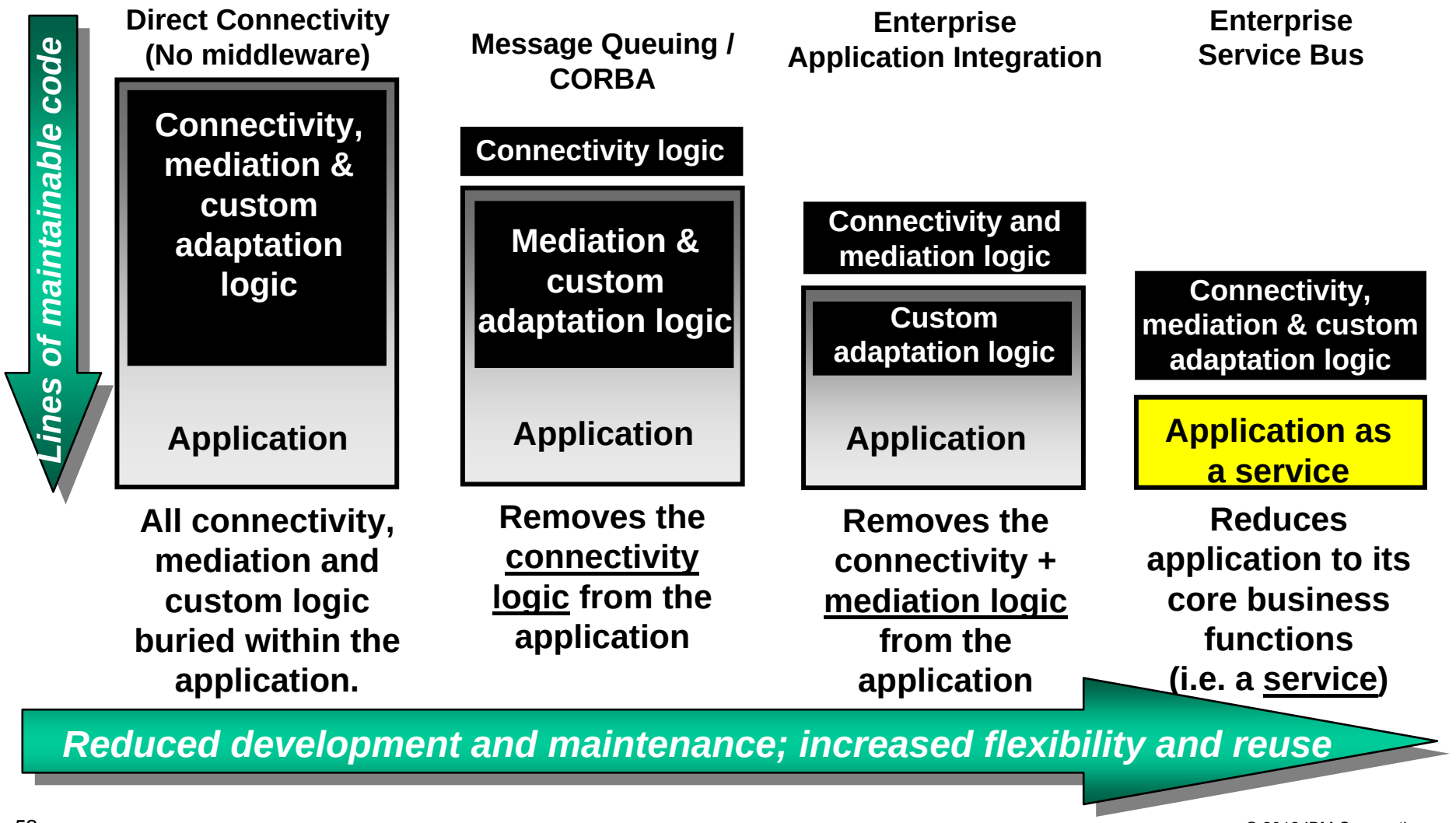
ESB (Enterprise Service Bus) – Service Virtualization

- **ESB acts as an intermediary (proxy) between requestor and provider**

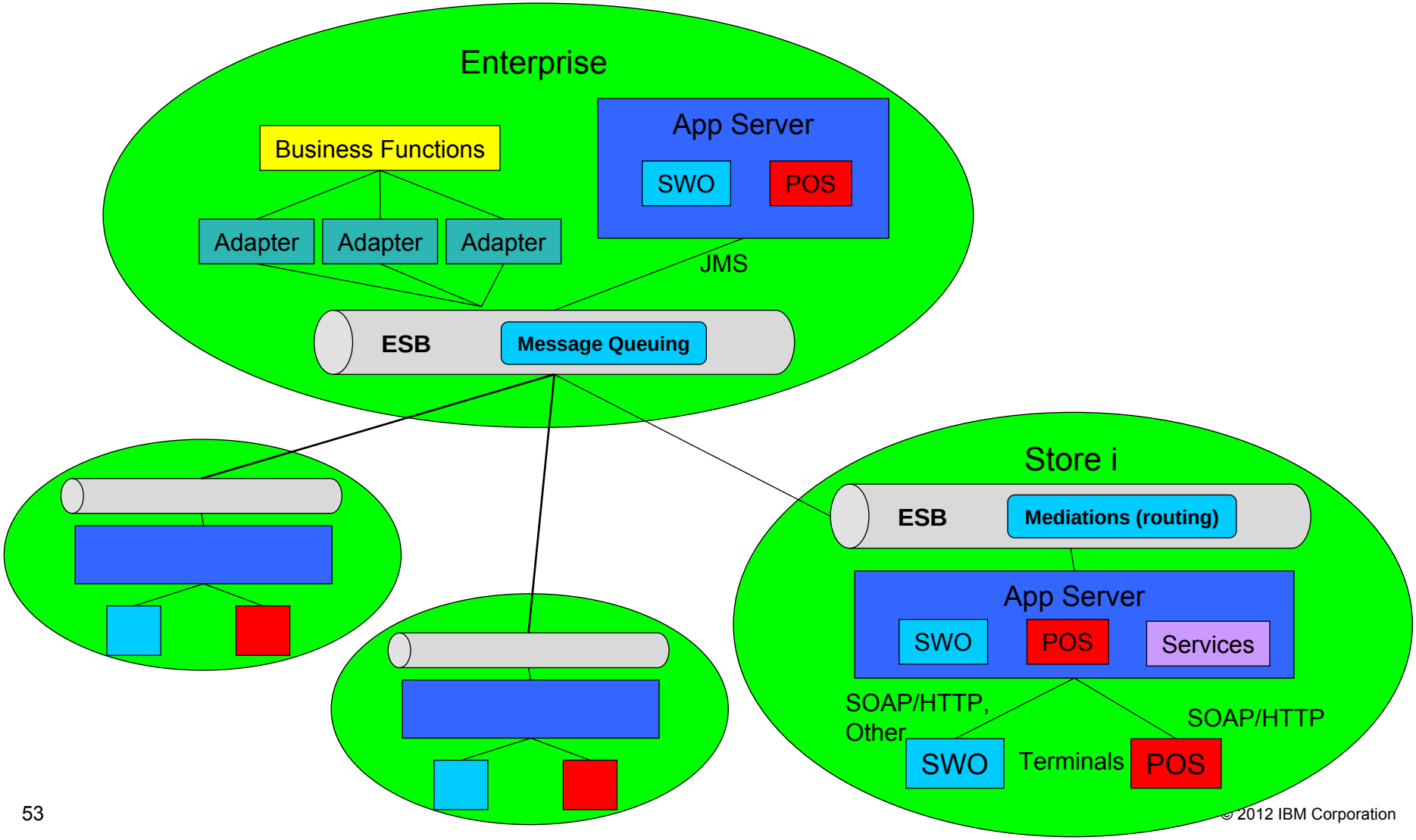


- **ESB provides *service virtualization* of**
  - ***Location and identity***
  - ***Interaction protocol***
  - ***Interface***
- **Interactions are *decoupled*, supporting *separation of concerns***

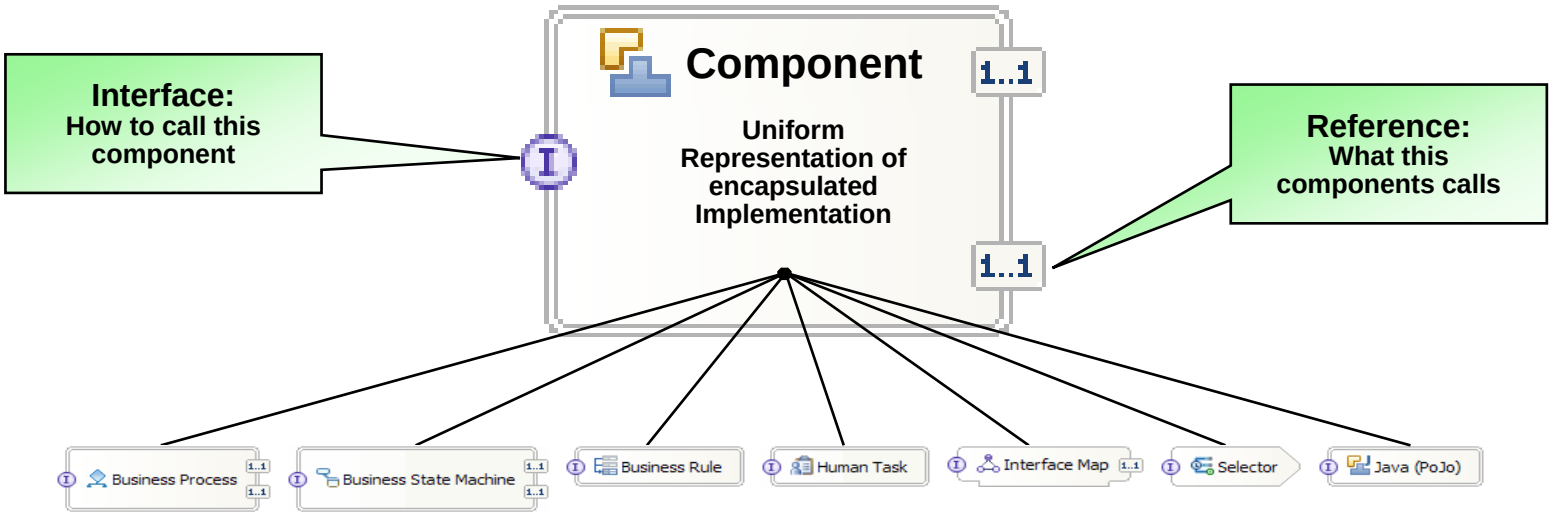
ESB is today's technology



ESB Pattern in Action – Retail Scenario



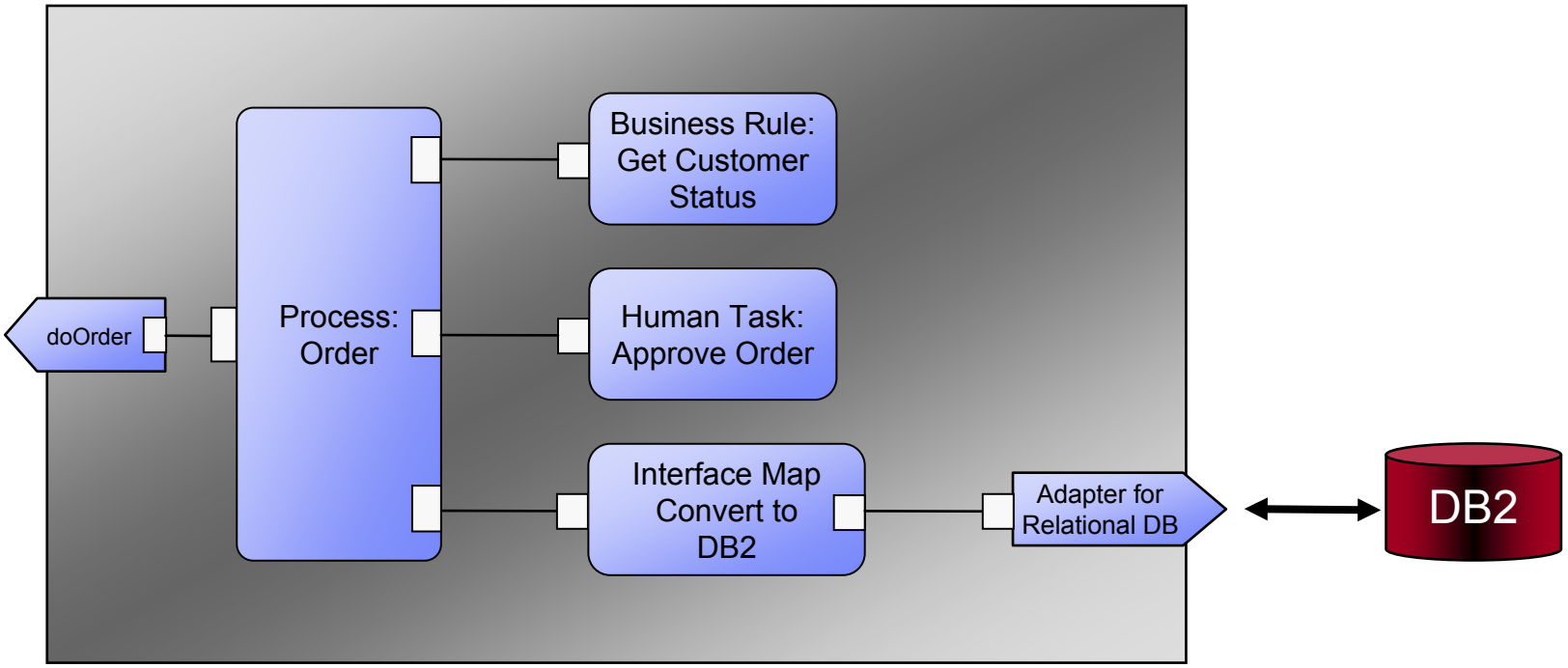
Standard SCA (Service Component Architecture) for Common Invocation



Encapsulate components for reuse

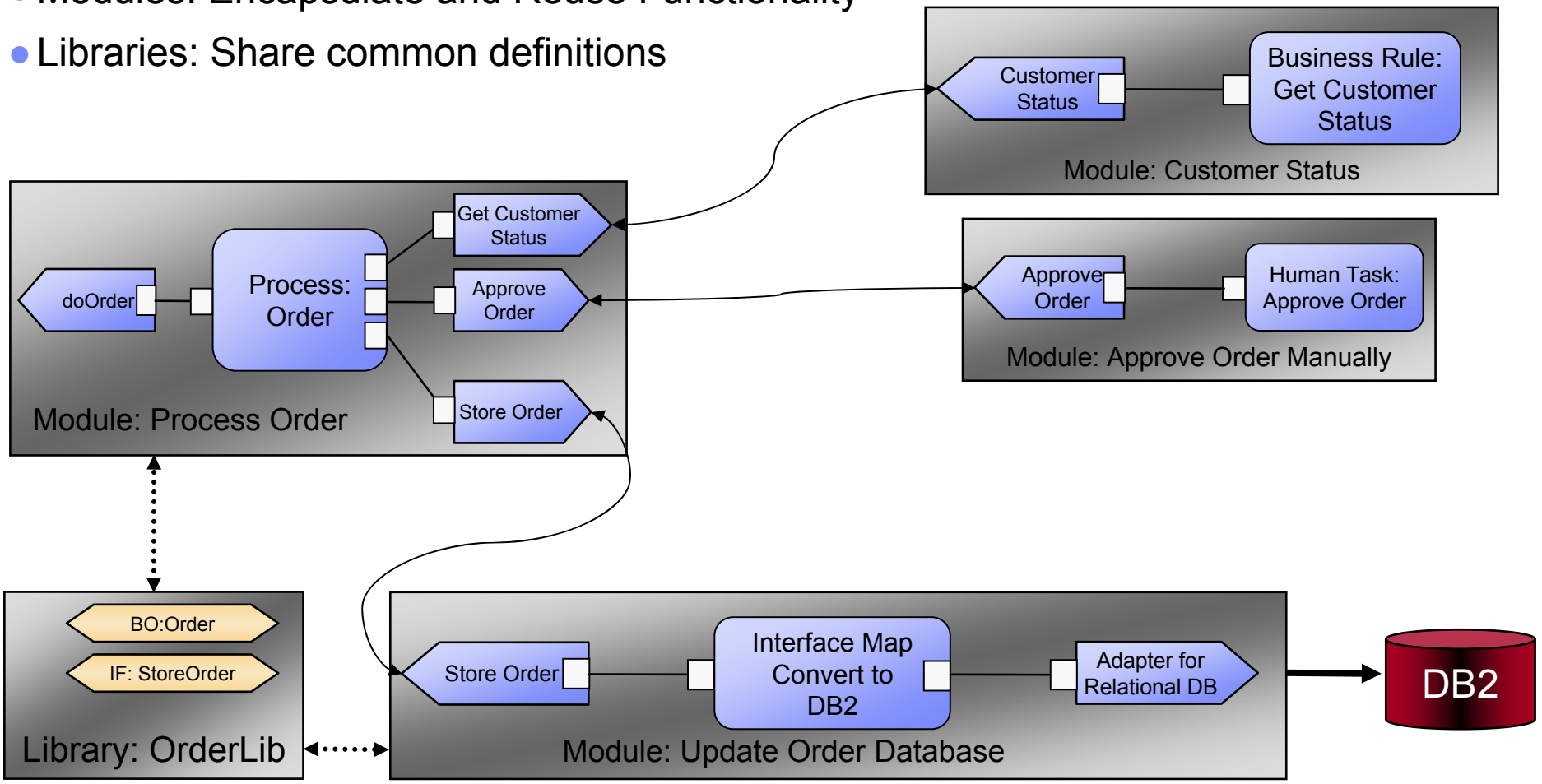
All components (e.g., services, rules, human interactions) are represented consistently and invoked identically

Standard SCA (Service Component Architecture) – Component Assembly



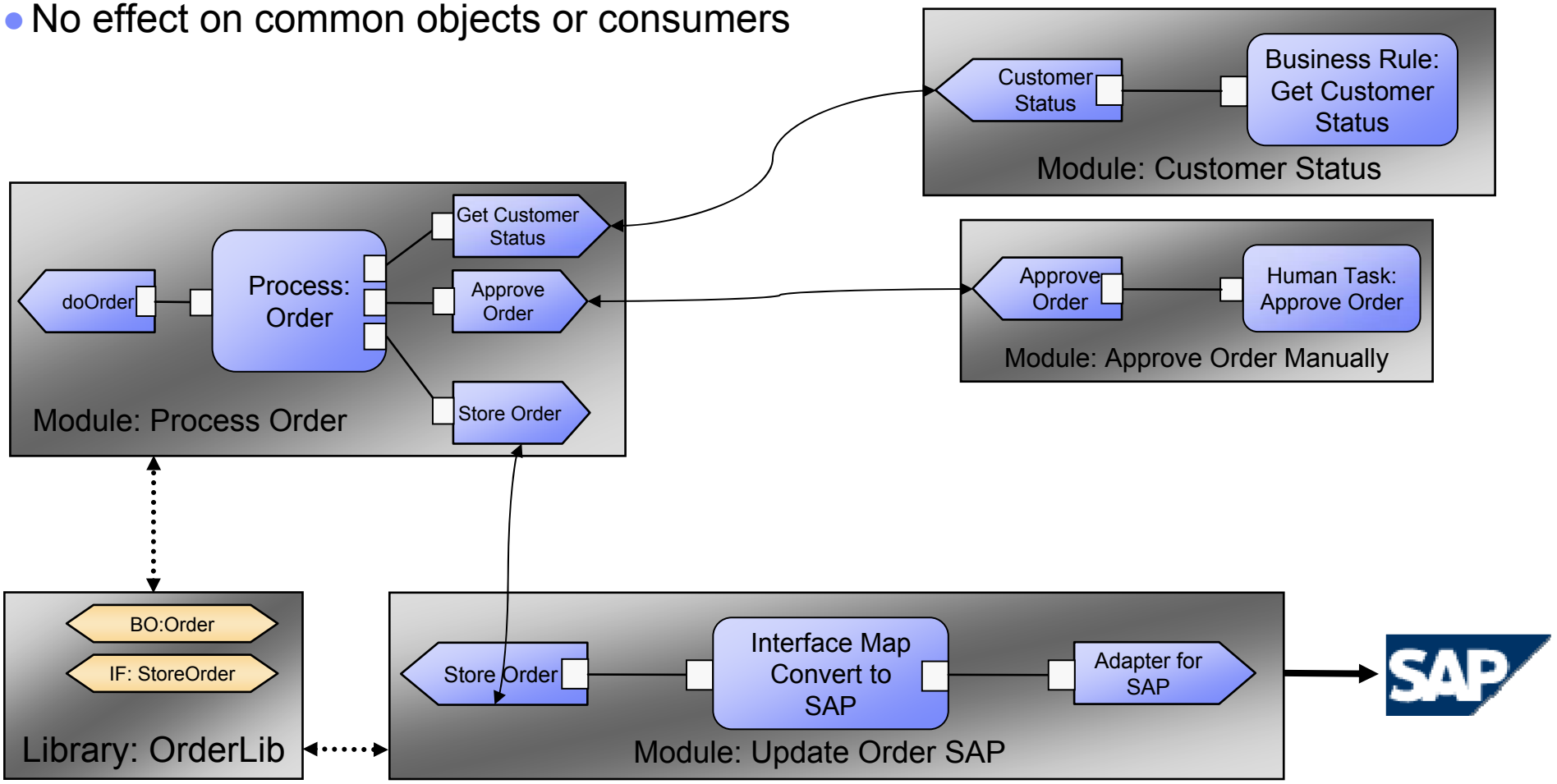
SCA (Service Component Architecture) – Example Part 1

- Modules: Encapsulate and Reuse Functionality
- Libraries: Share common definitions

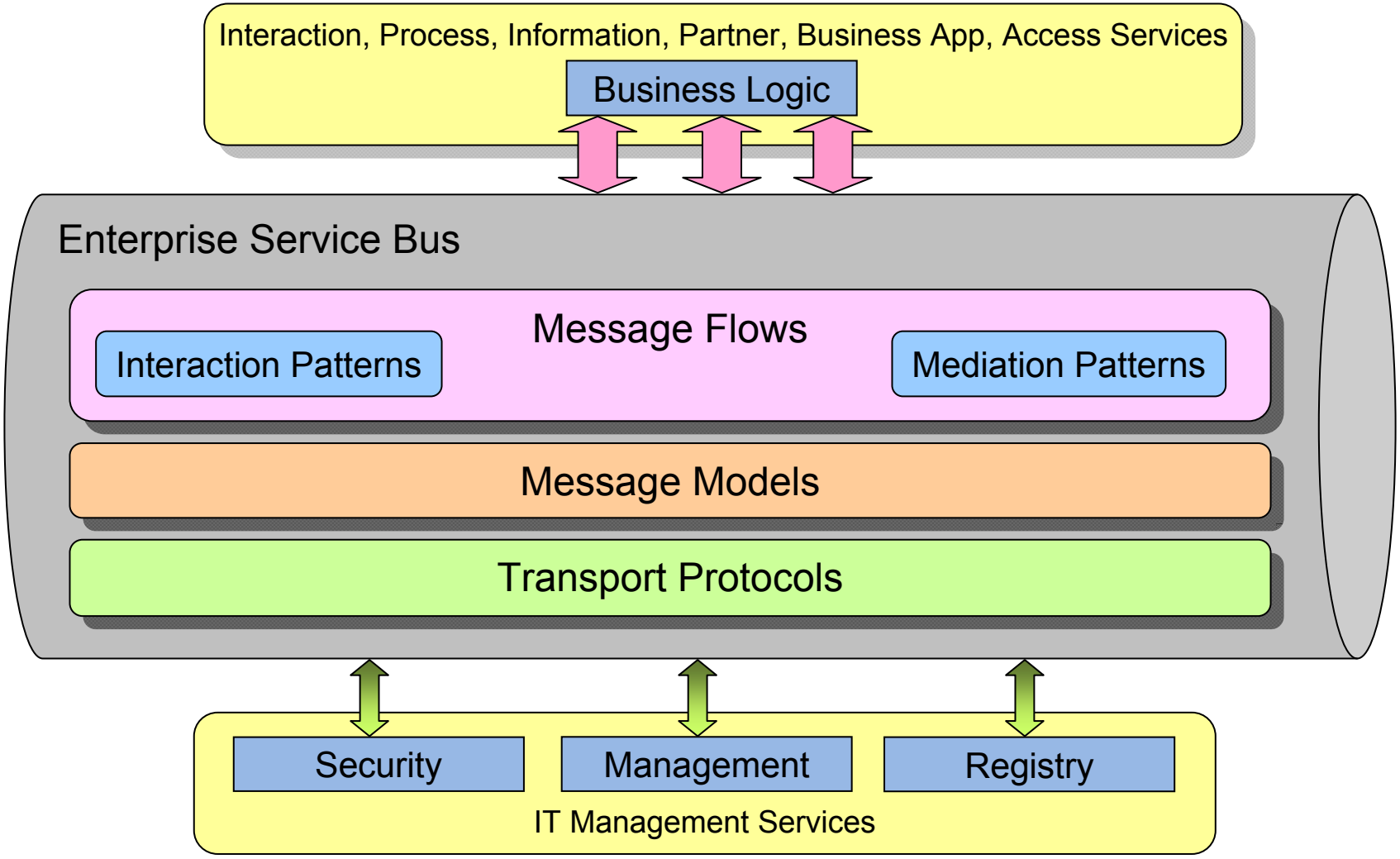


SCA (Service Component Architecture) – Example Part 2

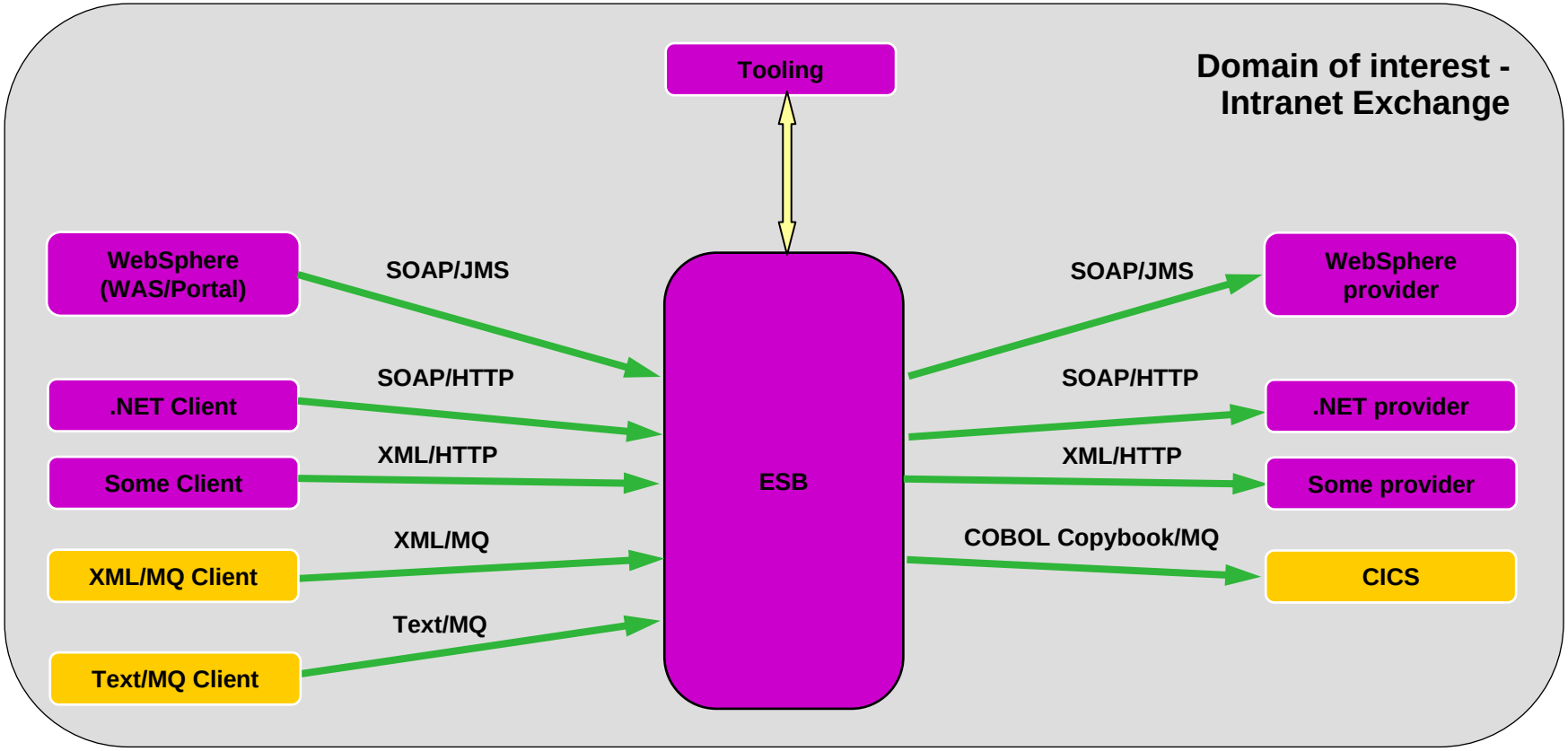
- Store Order in SAP instead of DB2
- No effect on common objects or consumers



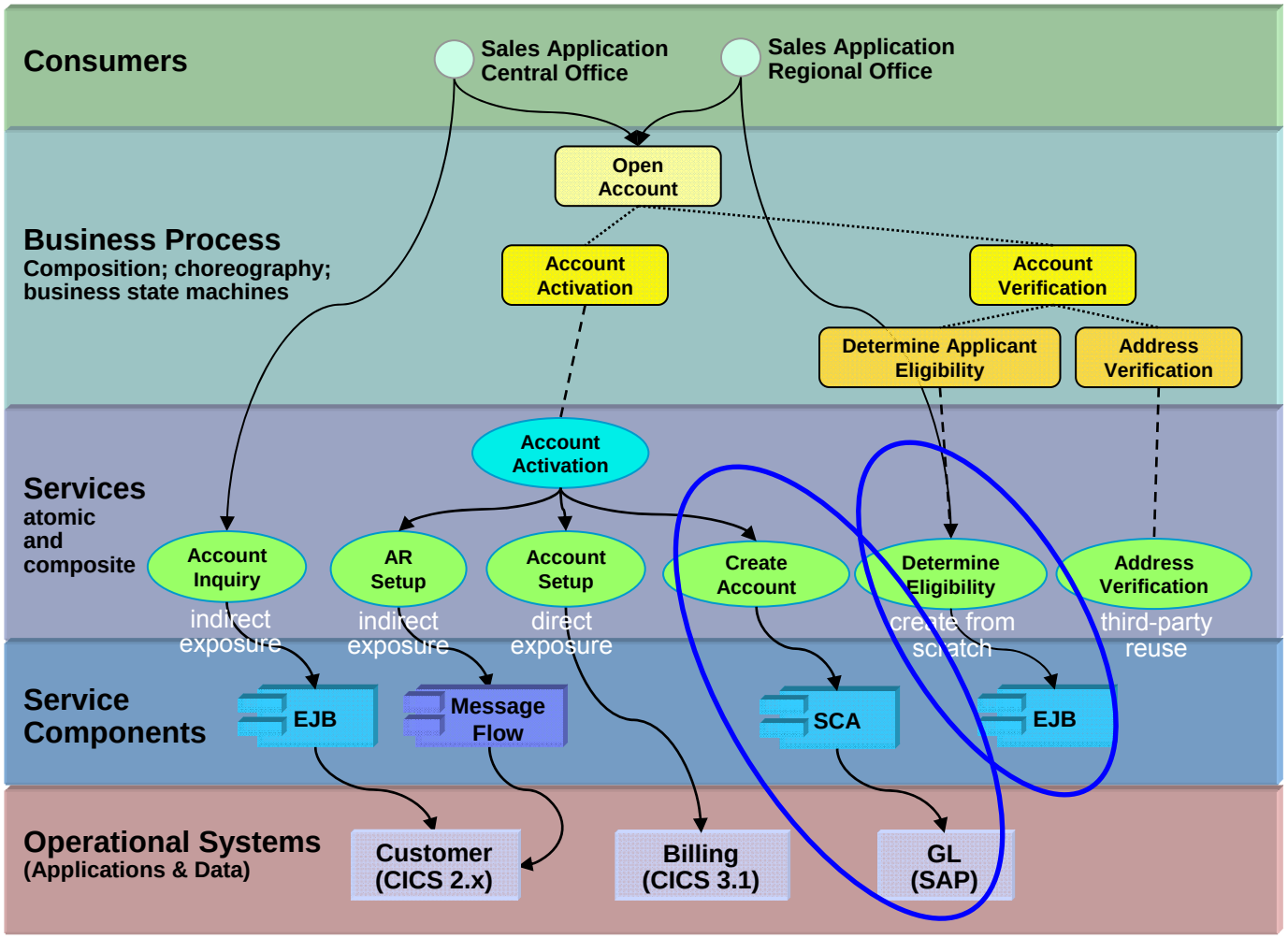
Expanded View of the Enterprise Service Bus



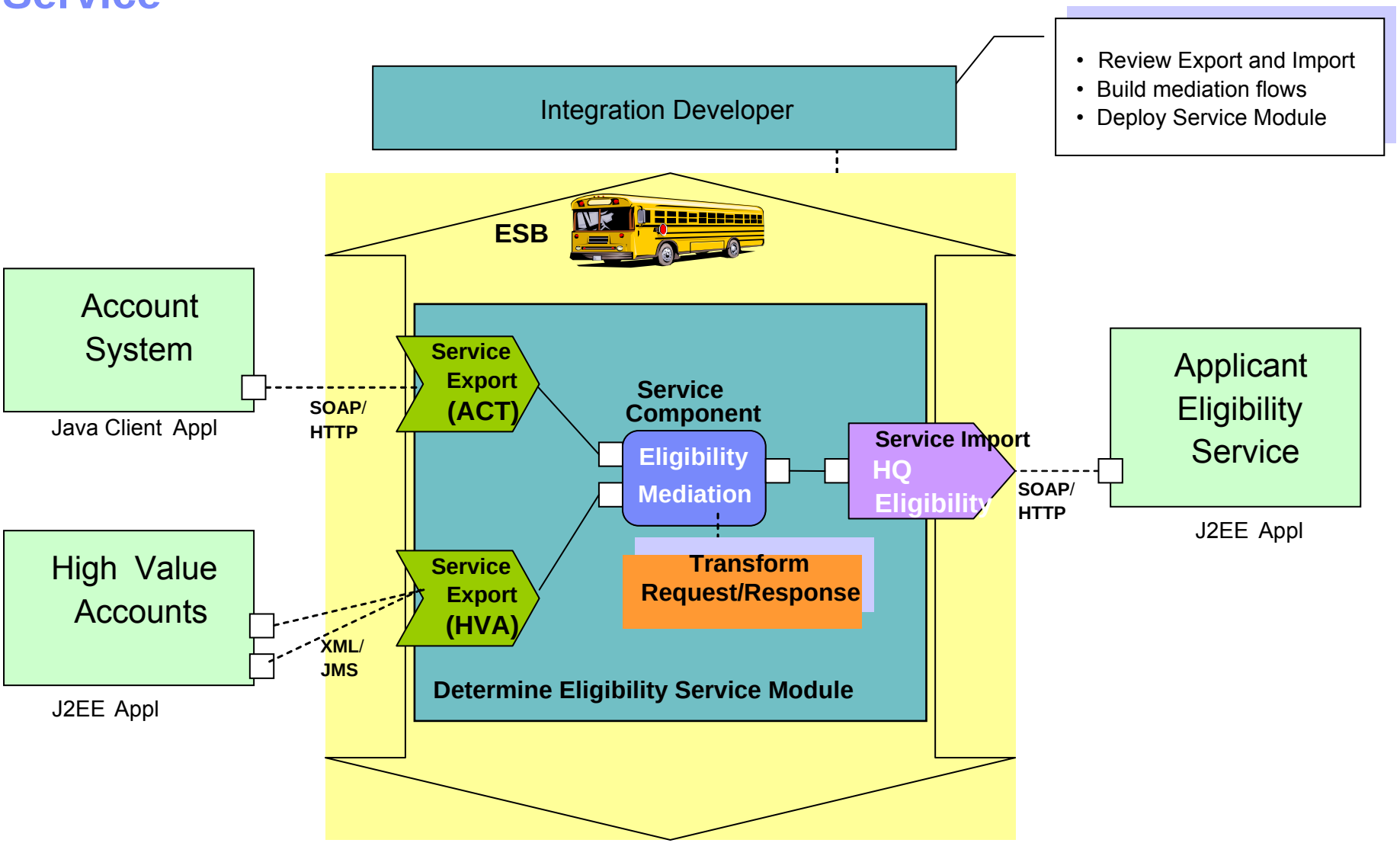
ESB – Multi-protocol Exchange – Intermediary decoupling heterogeneous consumers and suppliers



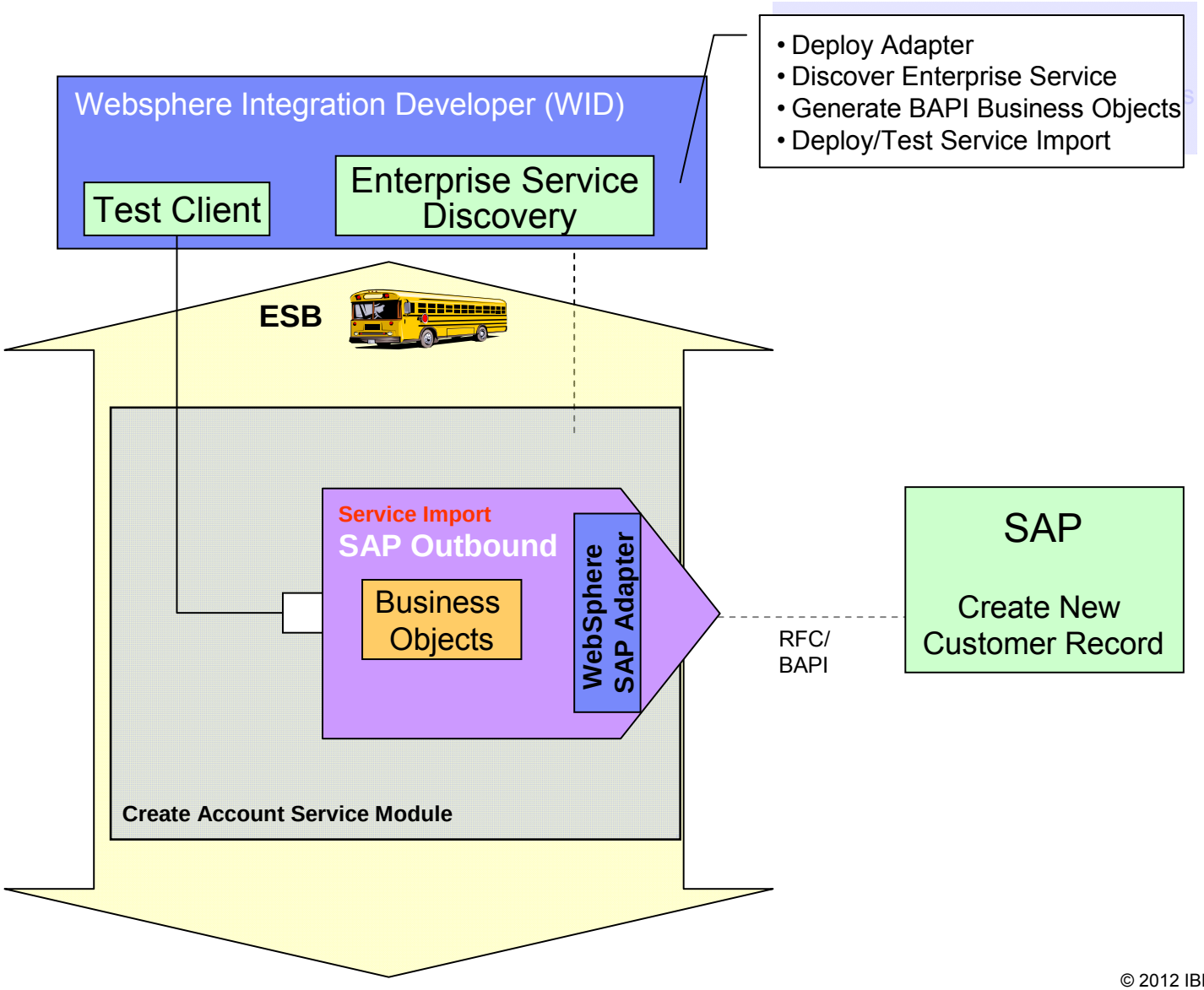
Example of ESB use



Example A of ESB use: Multiple Channel Access to Backend Service

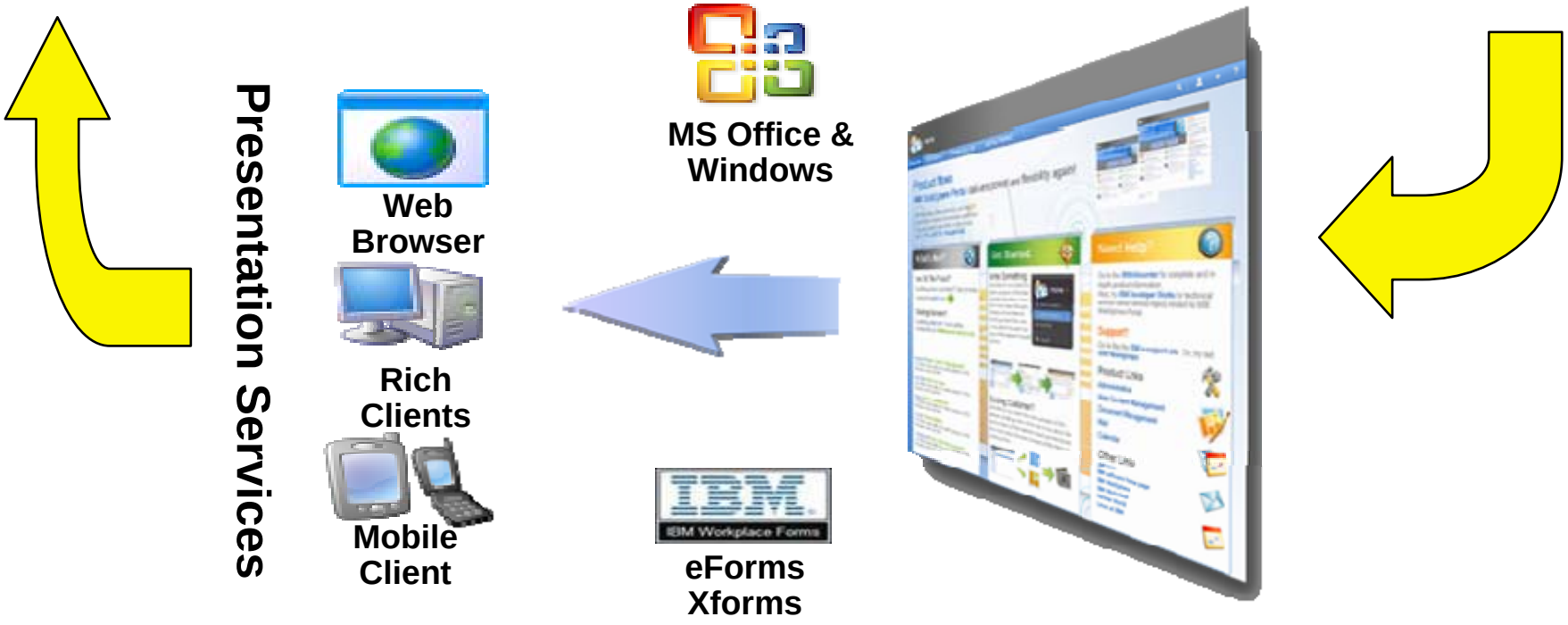


Example B of ESB Use: Create SAP Service

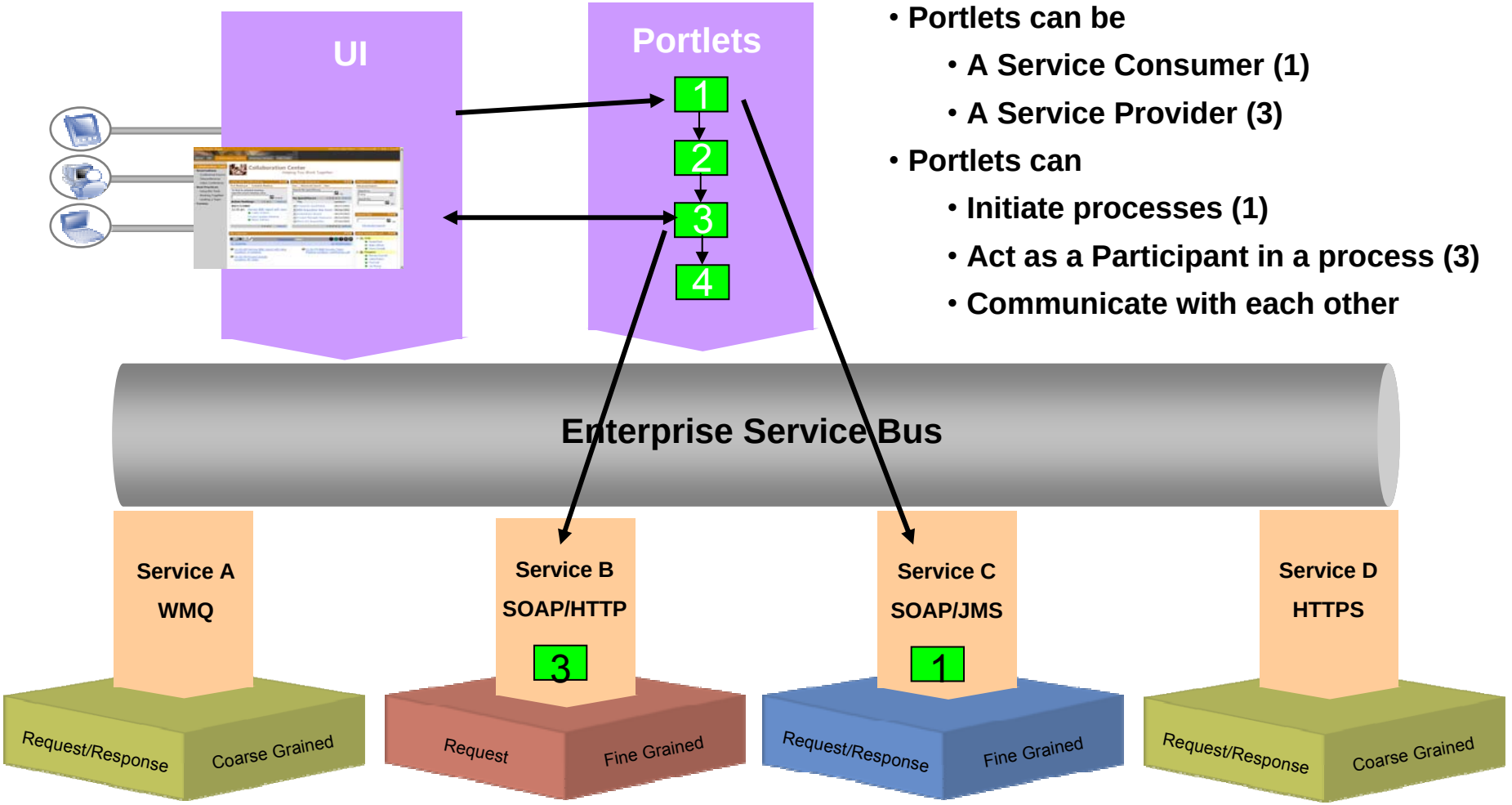


Interaction Services

Interaction Services: Using Portal As the "Front End" of SOA



What is an *Interaction Service*?



- Portlets can be
  - A Service Consumer (1)
  - A Service Provider (3)
- Portlets can
  - Initiate processes (1)
  - Act as a Participant in a process (3)
  - Communicate with each other

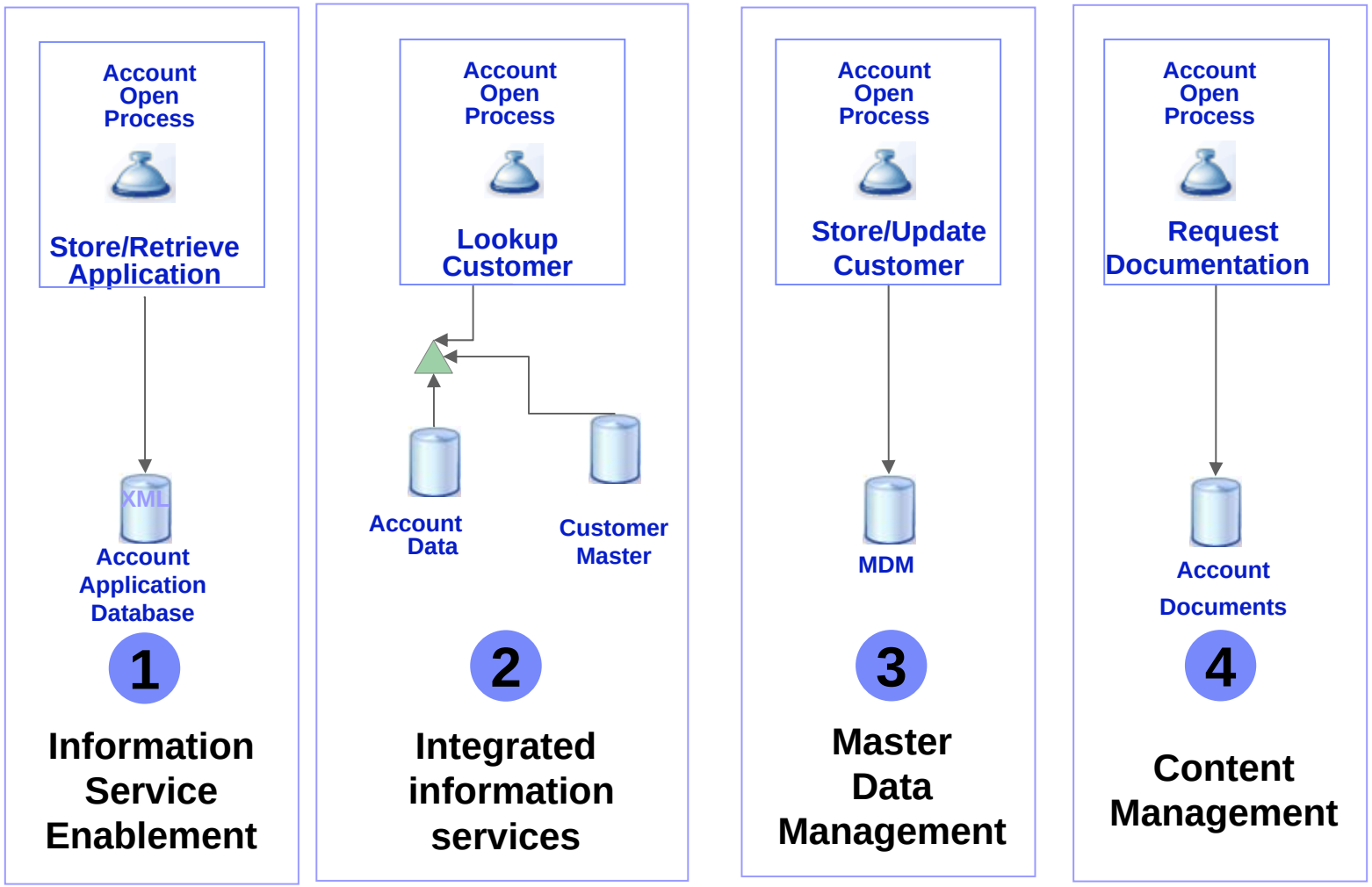
The Portal Framework Provides Service Aggregation

Information Services

Information Services in SOA Reference Architecture

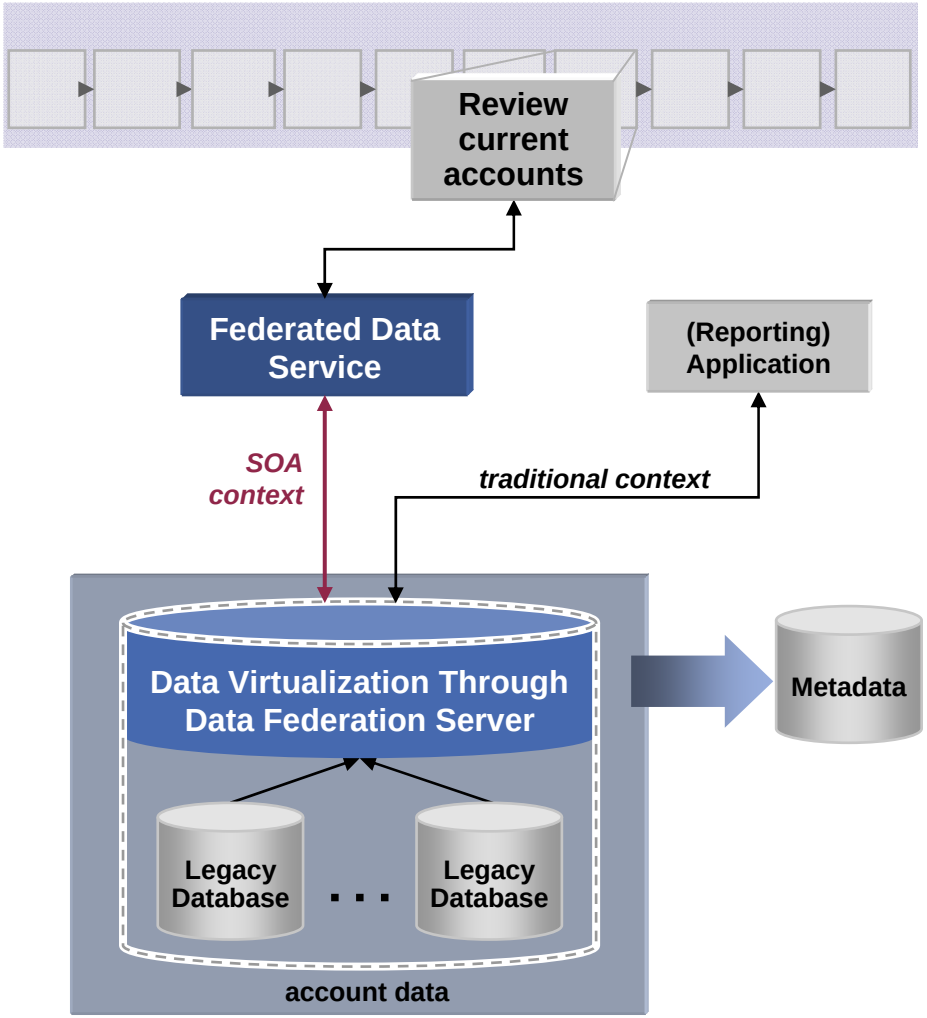
- **Delivering actionable information to people and processes**
- **Connect, enhance and deliver in-context information across diverse operating systems, applications and legacy systems through reusable services**
- **The Information Services enables consistent views and maintenance of data and content, providing a "single view of the truth" to people and processes**

Information Services: Several Patterns



Information Services: Pattern – Deliver Your Data *Virtualized Through Services*

- **As-Is Environment**
  - Data resides in disparate sources
  - Manual & redundant integration of data by multiple consumers results in high costs and inconsistent/inaccurate data
  - Slow response time due to inefficient real-time access
- **Solution Characteristics**
  - On demand integration instead of redundant data
  - Transparent & optimized access to distributed, heterogeneous sources
- **Results**
  - Real-time access to distributed information, fast response time
  - Scalable approach for adding more data sources



Closing Remark

Just remember – the future might bring more than you think

"I think there is a world market
for maybe five computers."
Thomas Watson, chairman of IBM, 1943

"Computers in the future may weigh
no more than 1.5 tons. "
Popular Mechanics, 1949

"There is no reason anyone would
want a computer in their home. "
Ken Olsen, founder of DEC,1977

"Prediction is difficult, especially
about the future"
Niels Bohr, 1957

"640K ought to be enough
for anybody. "
Bill Gates, 1981