What's in a Service?

Mike P. Papazoglou
INFOLAB,
Tilburg University,
The Netherlands
email: mikep@uvt.nl
http://infolab.uvt.nl/people/mikep

INFOLAB

UNIVERSITEIT VAN TILBURG

Agenda

- *Introduction*
- *The Multiple Facets of a Service*
- *Service Oriented Architecture*
- *Summary & Pointers to Open Research Problems*

Introduction

- Software services are self-contained, platform-agnostic computational elements that support rapid, low-cost and easy composition of loosely coupled distributed software apps.

- Services are *described*, *published*, *discovered*, and can be *assembled* to create complex service-based systems and applications.
  - They help integrate applications *not written with the intent to be easily integrated* with other applications & define architectures to *build new functionality while integrating existing application functionality*.

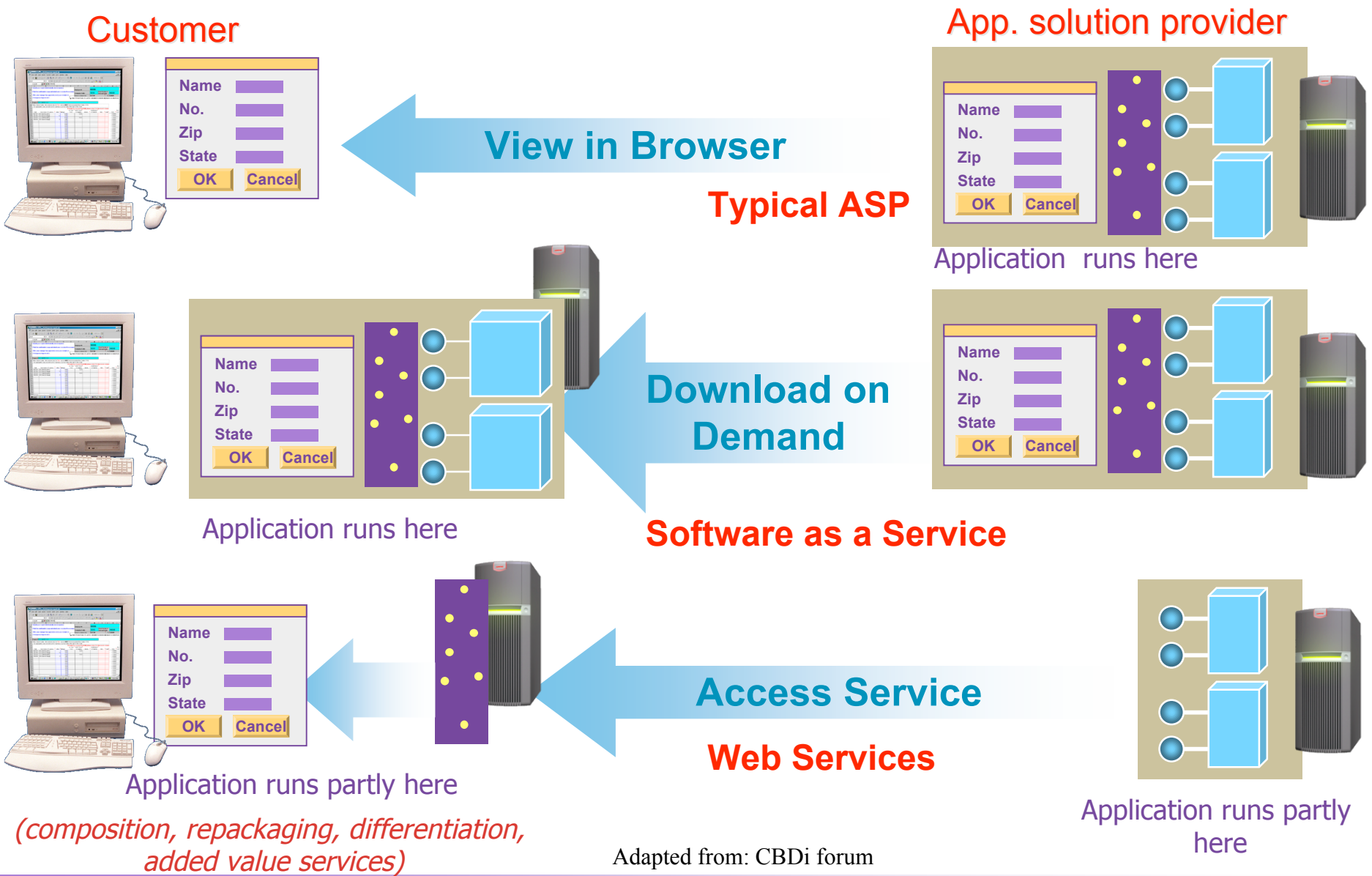
- Service-based applications are developed as *independent sets of interacting services* offering well-defined interfaces to their potential users using the principle of *loose coupling*.

Introduction (cntd)

- *Web services* share the characteristics of more general services:
  - expose their features programmatically over the Internet (or intra-net) via standard XML-based languages & protocols, and are implemented via a self-describing interface based on open Internet standards.

- Web services can vary in function:
  - from simple requests, e.g., credit checking and authorization, pricing enquiries, inventory status checking, or a weather report.
  - to complete business applications that *access & combine* info. from multiple sources, e.g., an insurance brokering system, an insurance liability computation, a package tracking system, etc.

Are ASP, Software as a Service, & Web Service the same?



Adapted from: CBDi forum

Evolving to Service Oriented Computing

- *Requires novel architectures & infrastructure to address challenges*
- Service proliferation drives the need for
  - Business & technical Services
  - Flexible & pro-active management
  - Comprehensive governance
  - Ease of composition and change
- Focus *shifting* to Business Processes:
  - The Business Process as Service
  - The Business Process as a Catalyst for Collaboration

Agenda

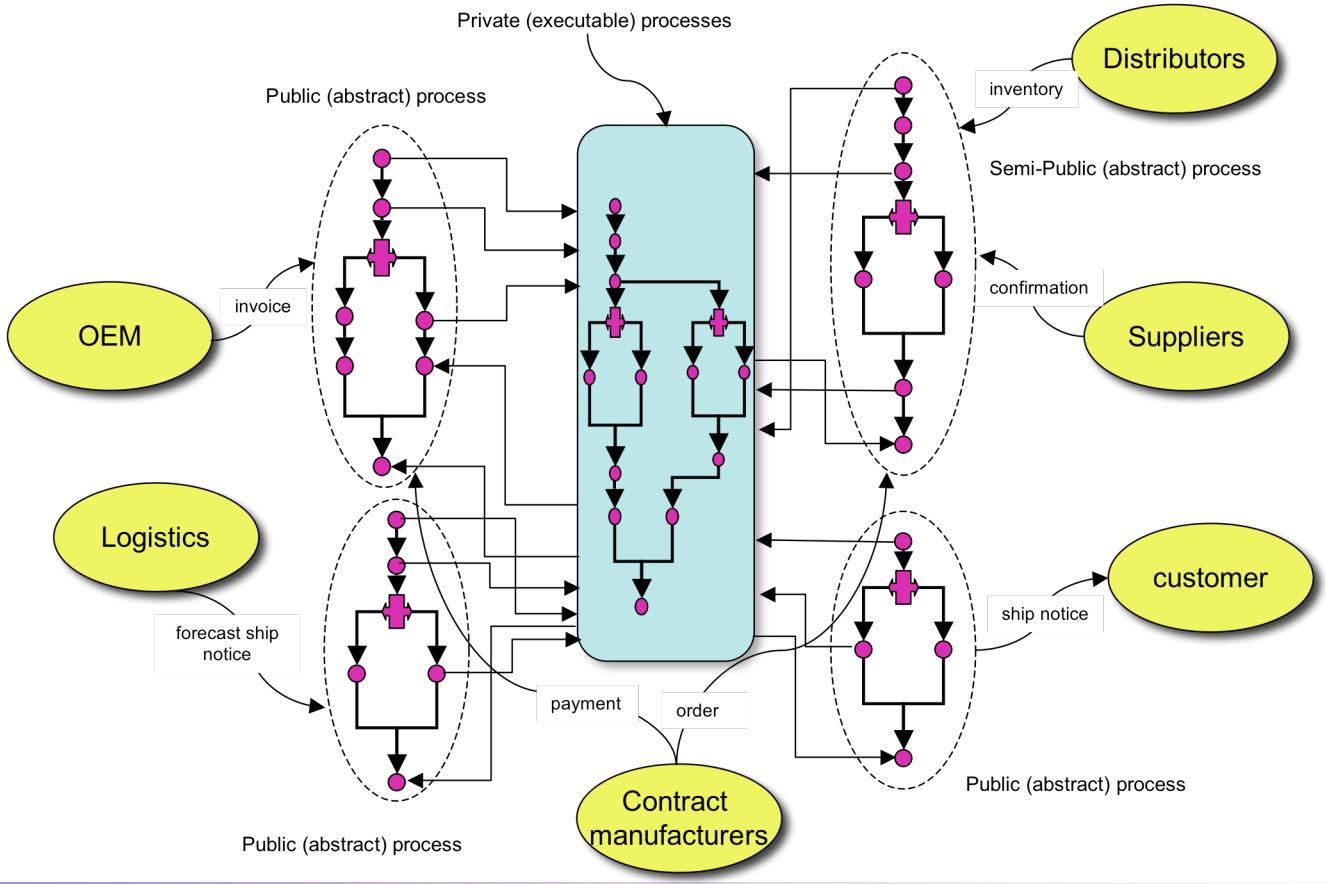
- *Introduction*
- *The Multiple Facets of a Service*
- *Service Oriented Architecture*
- *Summary & Pointers to Open Research Problems*

Types of a Service

- **Informational services**: simple in nature, provide access to content & multiple-data sources. Interact with an end-user by means of simple request/response sequences (*stateless* & *synchronous*):
  1. Pure content services: give programmatic access to content, e.g., financial info., stock quote info., design info., news items, etc.
  2. Seamless aggregation services: seamless aggregation of info. across disparate data sources including back-end systems,
     - e.g., logistic services that constitute the actual front-ends to large # of distributed data sources.
  3. Information syndication services: data-centric offered by a 3rd-party & run the whole range from commerce-enabling services, e.g., logistics, payment, & tracking to other value-added commerce services, such as rating services.

Types of a Service (cntd)

- ***Complex services*** *involve the assembly and invocation of many pre-existing services, possibly in diverse enterprises, to complete a multi-step business interaction (i.e., business process), e.g., a supply-chain application involving an Original Equipment Manufacturer (OEM) - stateful & asynchronous.*



Service Façades

- A service should contain minimally four facade elements:
  - Structural façade: defines the service types, messages, interfaces & operations (*service signature*).
  - Behavioral façade: entails understanding the effects & side effects of service operations & semantics of input & output messages, e.g., how can we cancel or update an order.
  - Policy façade: *describes policy assertions and constraints* & QoS considerations on the services between interacting parties. It:
    - prescribes, limits, or specifies any aspect of a business agreement.
  - Vocabulary and best practices façade: includes definition of:
    - common business processes to achieve canonization/standardization of processes and services;
    - common data-interchange formats i.e., standard messages/protocols that are exchanged in the context of processes/transactions; and,
    - common terminology at the level of data items and messages to bridge varying service terminologies.

- Provisioning and Charging Model: service providers must come up with viable business models that address factors such as business service metering, rating and billing.
- Operational Categories of Services: services can be classified wrt their operational characteristics:
  1. Business services: automate a generic business task with significance to the business.
  2. Technical services: coarse-grained services that provide the technical infrastructure enabling the development, delivery, maintenance & provisioning of singular business services & their integration into processes. They also provide capabilities that maintain QoS.
  3. Fine-grained utility (or commodity) services, which provide value to & are shared by business services across the organization.
     - services implementing calculations, algorithms, directory management services, etc.

Agenda

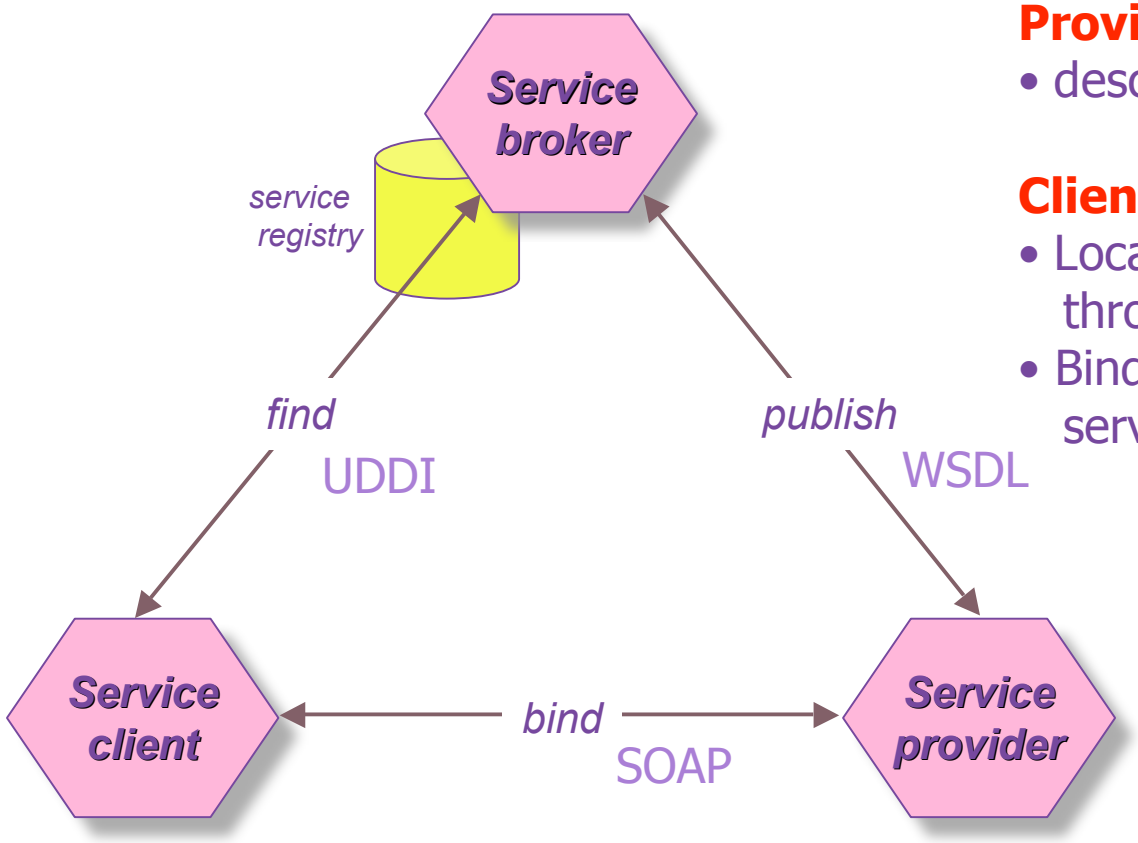
- *Introduction*
- *The Multiple Facets of a Service*
- *Service Oriented Architecture*
- *Summary & Pointers to Open Research Problems*

Service Oriented Architecture

- SOA is an architectural approach to *loosely coupled, protocol independent, standards-based distributed computing* where software resources available on a network are considered as services.
- The service is designed in such a way that it can be *invoked by various service clients* & logically decoupled from any service caller.
- SOA creates *service level abstractions* that map to the way a business actually works.
- *Leverages investment* in existing application/resource assets.
- *Creates composite applications* which partially automate business functions.
- An SOA is based on the combination of and interaction between services, associated with messages & *governed by policies*.

SOA Roles & Functions

Although SOAS can be implemented using different technologies, e.g., middleware technologies like J2EE, JMS,
Web services is the preferred environment for realizing the SOA promise of maximum service sharing, reuse, and interoperability.

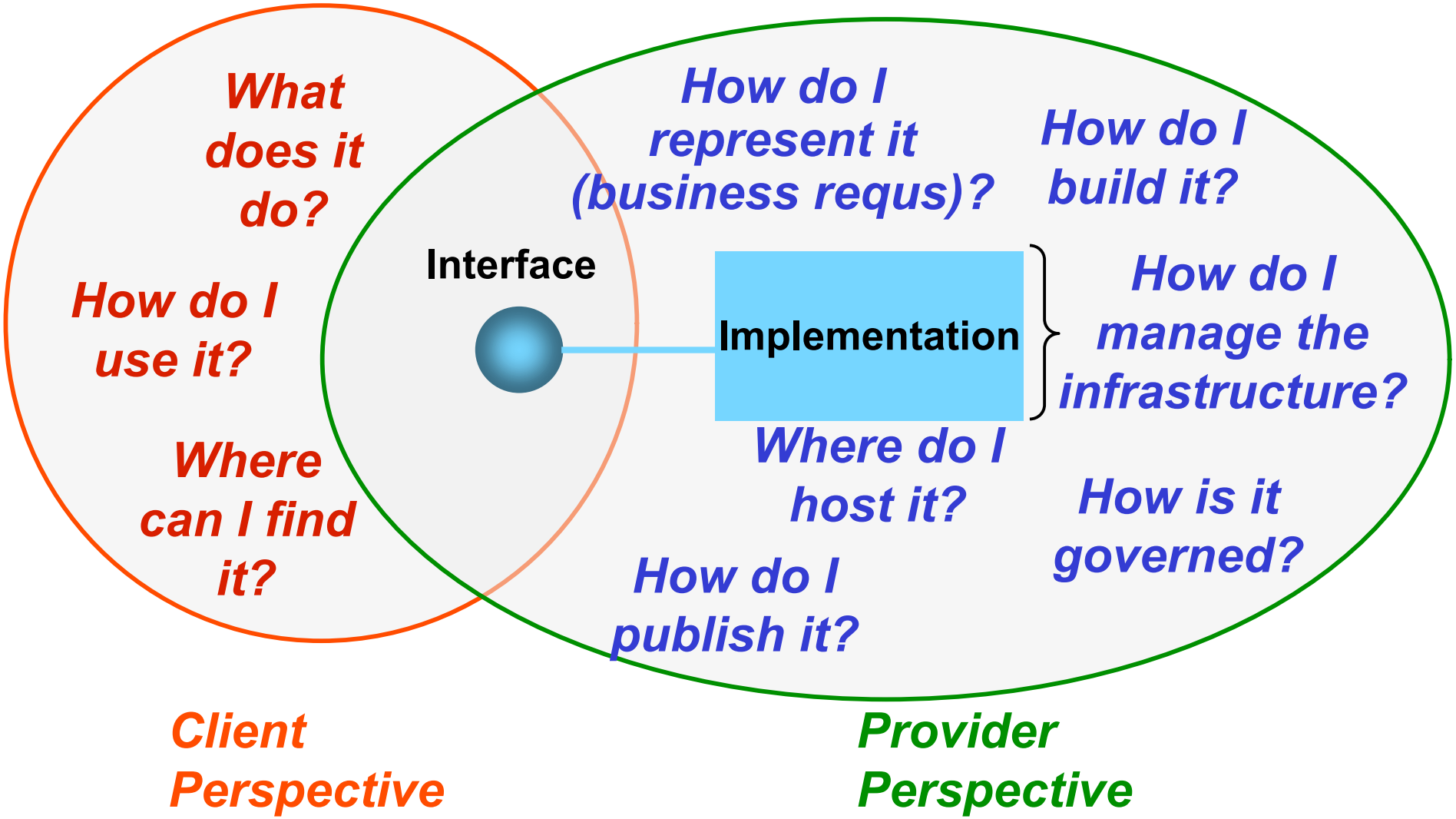


Provider:
- describes & publishes services

Client:
- Locates service provider through service registry
- Binds to & invokes service based service interface

SOA is focused on creating:
- a design style,
- technology, and
- process framework

SOA Concerns



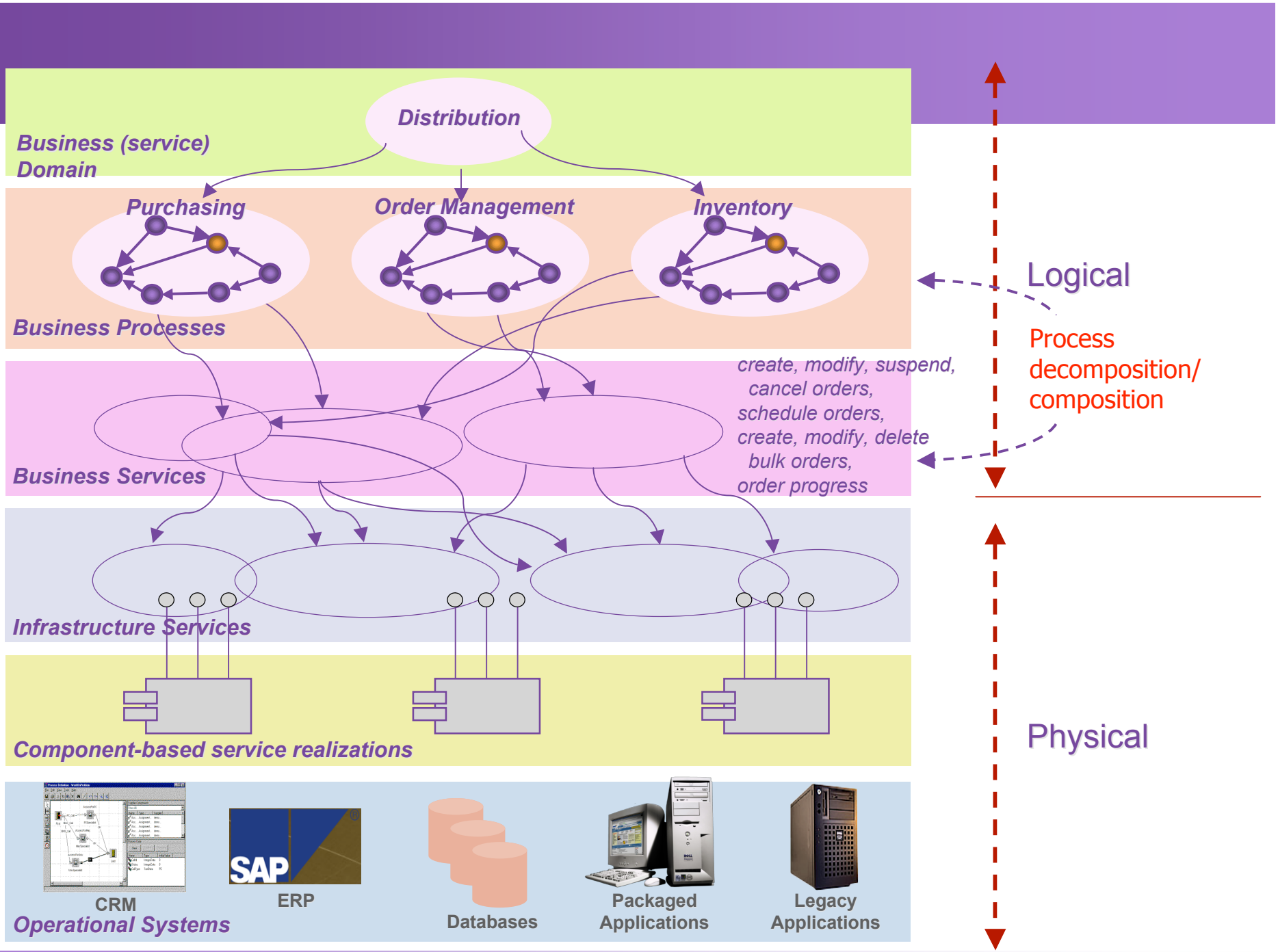
Service Oriented Architecture (cntd)

The following key SOA enablers form the core of (inter- and cross)-enterprise integration:

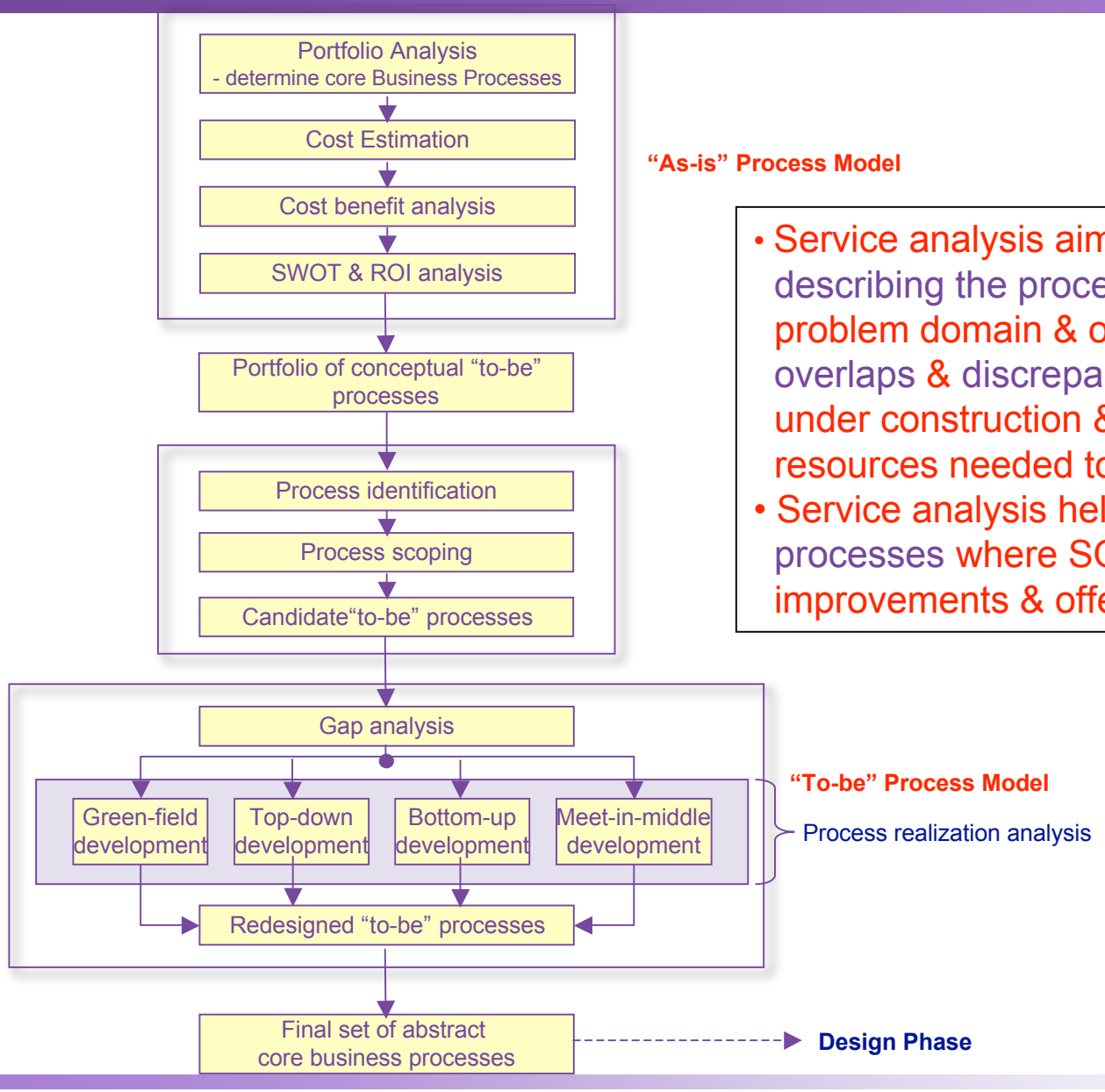
1. Planning & engineering the SOA,
2. SOA implementation,
3. SOA management and monitoring, and
4. Governing the SOA.

1. Planning & Engineering the SOA

- A service-oriented approach to solutions development is needed to produce an SOA in a way that it aligns with business process interactions between partners to accomplish a common business goal & stated functional and non-functional business requirements.
- A service-oriented approach to solutions development requires a broader review of its impact on how SOA solutions are:
  - designed;
  - what it means to assemble them from disparate services; and
  - how deployed services-oriented applications can evolve and be managed.
- This requires addressing common concerns such as the identification, specification and realization of services, their flows and composition into processes, as well as the enterprise-scale components needed to realize (implement) them and ensure the required QoS.


Business (service) Domain
Distribution
Purchasing
Order Management
Inventory
Business Processes
create, modify, suspend, cancel orders, schedule orders, create, modify, delete bulk orders, order progress
Business Services
Infrastructure Services
Component-based service realizations
CRM
ERP
Databases
Packaged Applications
Legacy Applications
Operational Systems
Logical
Process decomposition/ composition
Physical

Service Analysis



- Service analysis aims at identifying & describing the processes in a business problem domain & on discovering potential overlaps & discrepancies between processes under construction & available system resources needed to realize services.
- Service analysis helps prioritize business processes where SOA can contribute to improvements & offer business value potential.

Service Design

- Service design requires developers to define related, well-documented interfaces for all conceptual services identified by the analysis phase, prior to constructing them.
- The design phase encompasses the steps of:
  - singular service specification,
  - business process specification, and
  - policy specification for both singular services and business processes.
- Service design is based on a twin-track design approach that provides two production lines – one along the logical part and one along the physical part of the SOA – and considers both functional and non-functional service characteristics.

Service Design: Service Granularity Concerns I

- **Service granularity** refers to the scope of functionality exposed by a service.
- Services may come at two levels of granularity:
  - A coarse-grained interface might be the complete processing for a given service, e.g., "SubmitPurchaseOrder"
    - the message contains all business information needed to define a purchase order.
  - A fine-grained interface might have separate operations for: "CreateNewPO", "SetShippingAddress", "AddItem", etc.
- Fine-grained services usually provide basic data access or rudimentary operations. These services are of little value to business applications.
- Coarse-grained services are composed from finer grained services.

Service Design: Service Granularity Concerns II

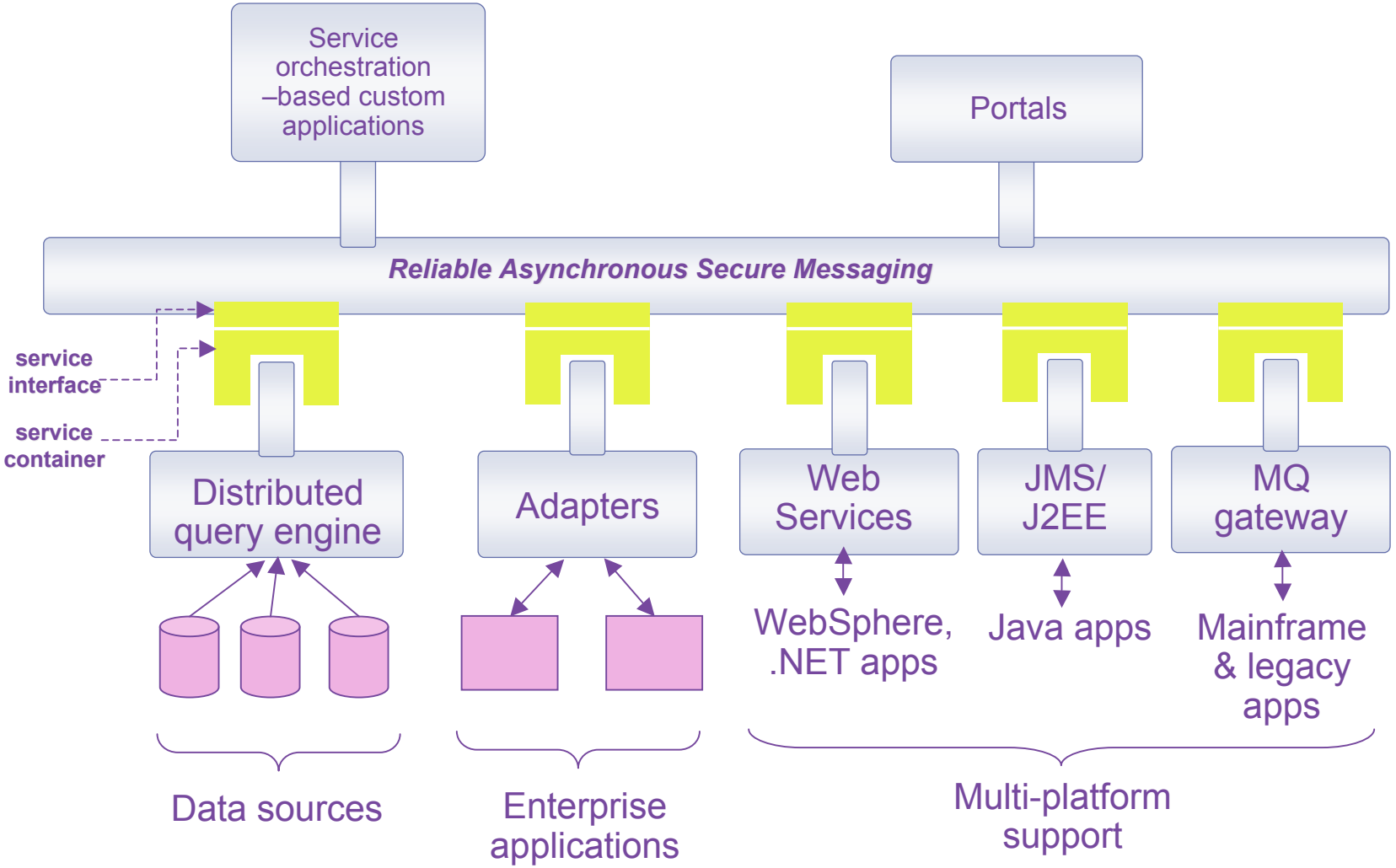
- The frequency of message exchange is an important factor. Sending & receiving more info. in a single request is more efficient than sending many fine-grained messages.

  - *Several redundant, fine-grained services lead to increased message traffic & tremendous overhead & inefficiency.*
  - *A small collection of coarser-grained services - each of which implements a complete business process - that are usable in multiple scenarios is a better option.*

- Heuristics identify the right level of granularity for services, e.g., clearly identifiable business concepts, highly usable and reusable concepts, concepts that have a high-degree of cohesion & low-degree of coupling.
- Vertical sectors, e.g., automotive, travel industry, etc, standardize business entities & processes by choosing their own levels of granularity.

2. SOA Implementation: Enterprise Service Bus

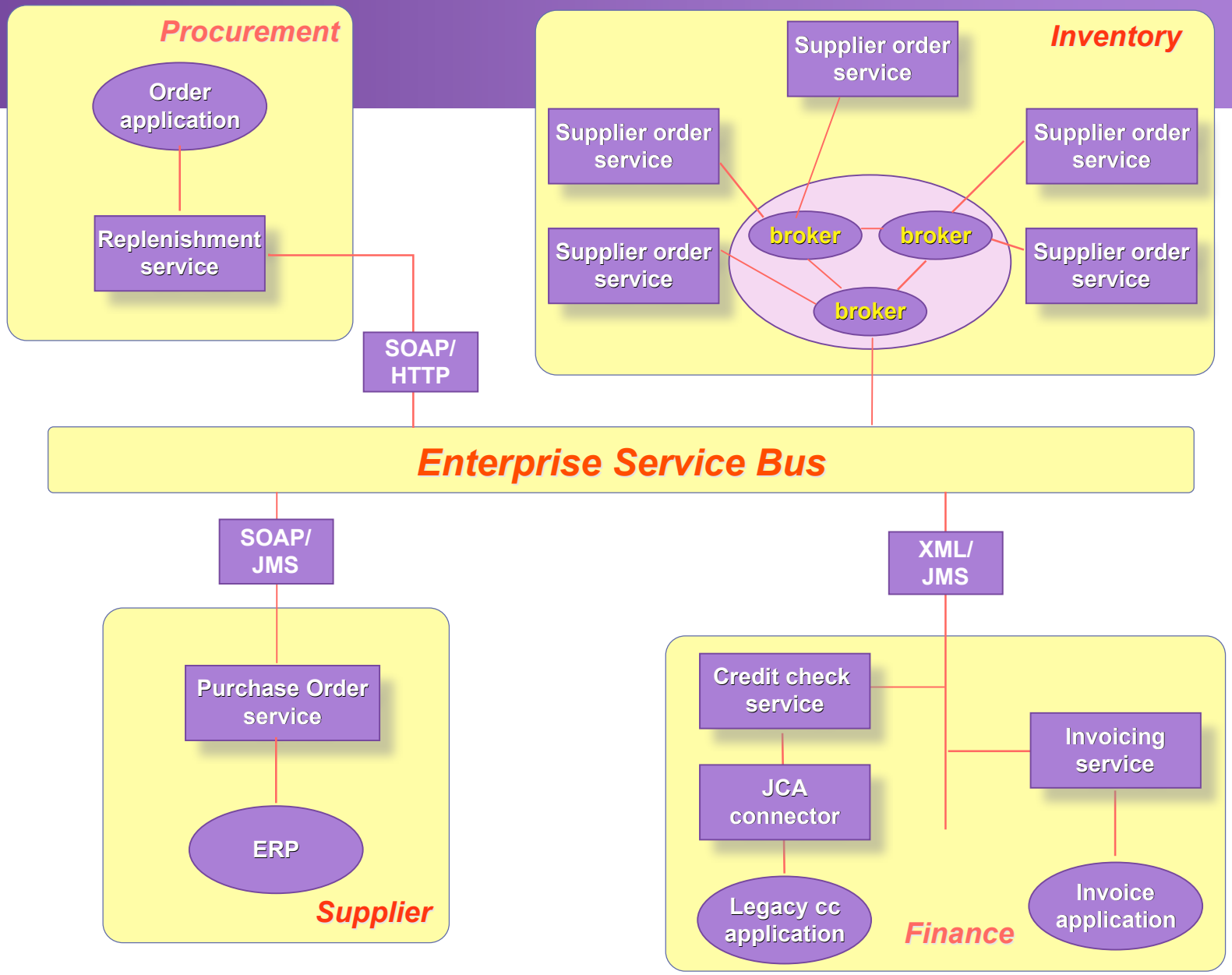
- ESB is a standards-based IT backbone that leverages Messaging Oriented Middleware functionality to connect heterogeneous systems.
- A way of building and deploying enterprise SOAs
- Combines features from different types of middleware into one package.
- Facilitates the deployment of Web services and solves integration problems.
- Key features:
  - Web Services: SOAP, WSDL, UDDI
  - Event-based, asynchronous delivery
  - Transformation
  - Routing: Publication/Subscribe, content-based, itinerary
- Platform neutral
  - Connect to anything in the enterprise: Java,.NET, legacy, WS-*, ..



Enterprise service bus connecting diverse applications & technologies

Key Capabilities of an ESB

- Service Communication Capabilities
  - ability route service interactions through a variety of protocols, and to transform from one protocol to another.
- Dynamic Connectivity Capabilities
  - ability to connect to Web services dynamically without using a separate static API or proxy for each service.
- Topic and Content-based Routing Capabilities
- Endpoint Discovery with Multiple QoS Capabilities
- Leveraging Existing (Legacy) Assets
- Integration & Transformation Capabilities
- Security Capabilities
- Business Process Orchestration
- Management & Monitoring Capabilities
- Scalability Capabilities



Scaling services in an enterprise service bus.

3. Services Management

- Composite service developments need mechanisms that provide insights
  - into the *health of systems* that implement services &
  - into the *status & behaviour patterns of loosely coupled applications*.
- **Services management** is the functionality required for discovering:
  - existence, availability, performance, health, patterns of usage, extensibility, control and configuration, life-cycle support & maintenance

  of services within the context of SOAs.
- Services mgt gathers info about the managed service platform, services and business processes and managed resource status & performance, (e.g., root cause failure analysis, SLA monitoring and reporting, service deployment, capacity planning, etc).

3. Services Management (cntd)

- Service management:
  - supports *active capabilities* versus traditional passive capabilities e.g., rather than merely raising an alert when a given service is unable to meet the performance requirements of a given service-level agreement, is able to take corrective action itself.
  - provides *global visibility of running processes*.
- Services manageability can distinguish between three functional parts:
  1. Services Monitoring: the ability to capture runtime and historical events from a particular service, for reporting and notification.
  2. Services Tracking: the ability to observe aspects of a single unit of work or thread of execution of services across multiple resources & enterprises.
  3. Services Control: the ability to alter the runtime behavior of a managed service.

Service Monitoring

Services monitoring focuses on:

- Infrastructure Monitoring: Distributed resources must be monitored for availability, performance and utilization so that developing problems can be quickly detected.
- Transaction Monitoring: when user experience degrades, individual transactions must be traced through the system to see which servers and which services are involved and where the problem lies.
- Resource Provisioning: When a Web services-based application is running careful monitoring of system activity can help to identify potential problems before they are manifested in overloads or failures.
- SLA Monitoring: an SLA reflects rigorous requirements for the key performance indicators for all services involved in it.
  - The SLA ties together systems and services from multiple groups & multiple vendors, defining the boundaries & operational characteristics.

Service Management & Monitoring

Services need to be managed in two dimensions:

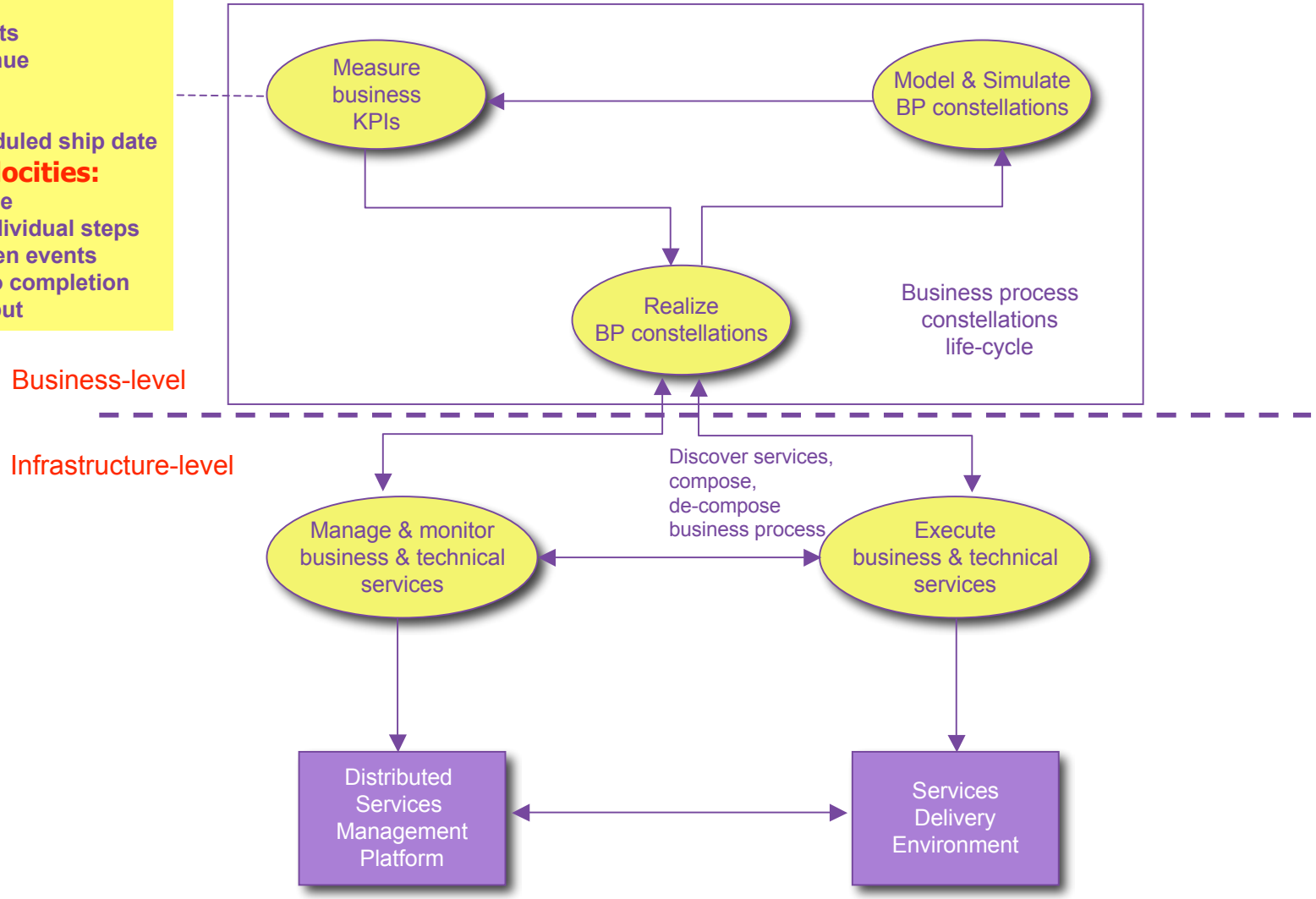
- The operational (or infrastructure) management dimension, in which a systems administrator starts and stops services and keeps track of how many instances of a Web service are running, in which containers, and on which remote systems.

- The tactical (or business) management dimension that provides a number of business activity monitoring and analytics capabilities that:
  - enable some human agent to watch over business operations,
  - diagnose problems as they occur so as to ensure that the services supporting a given business task are performing in accordance with service level objectives.

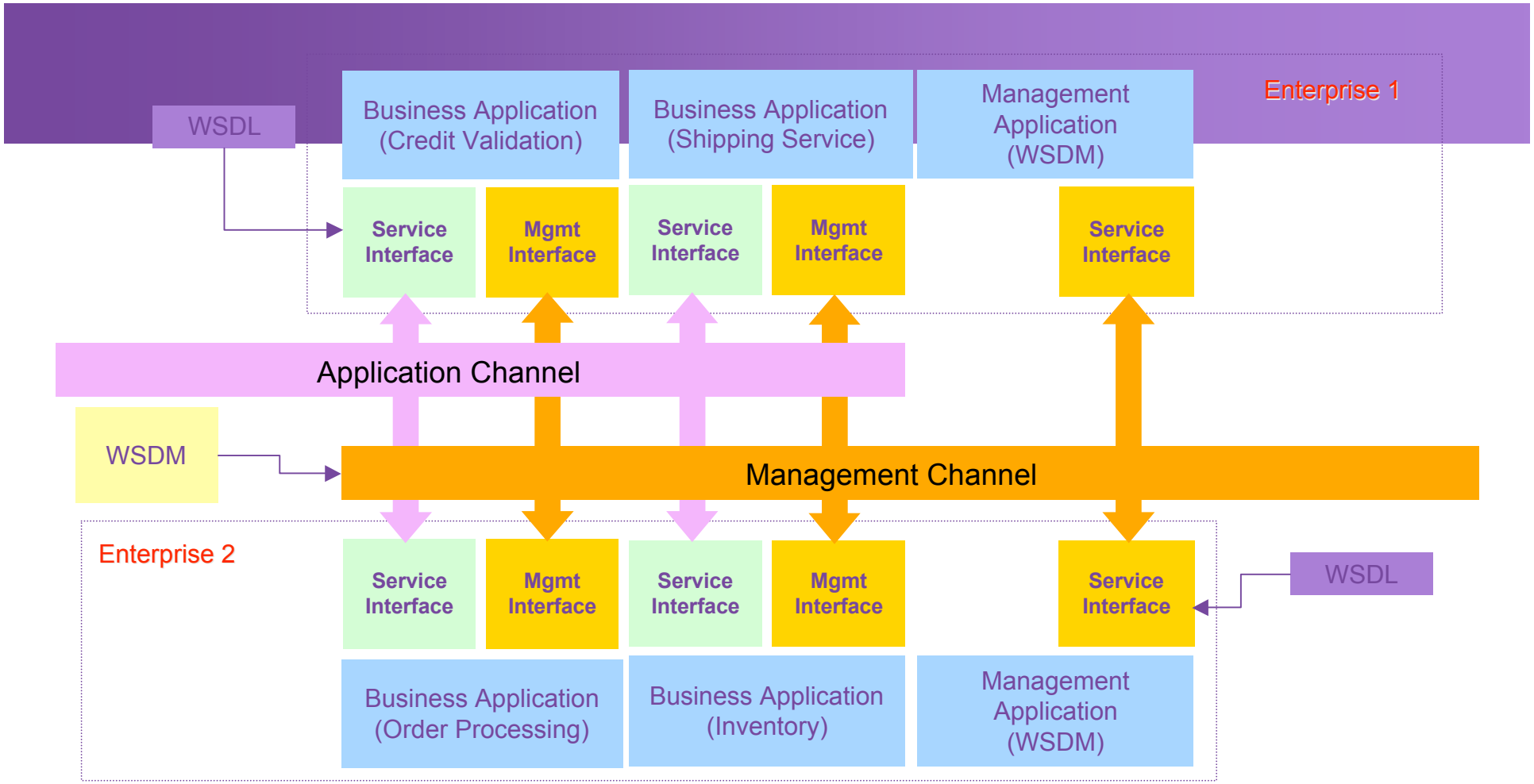
Combining SOA and BPM

Transaction volumes:
- # of transactions
- # of process events
- Transaction revenue
- Process revenue
- Costs & margins
- # of days to scheduled ship date

Transaction velocities:
- Process cycle-time
- Cycle-times of individual steps
- Wait-times between events
- Time remaining to completion
- Process throughput



The business and infrastructure levels for Web services management

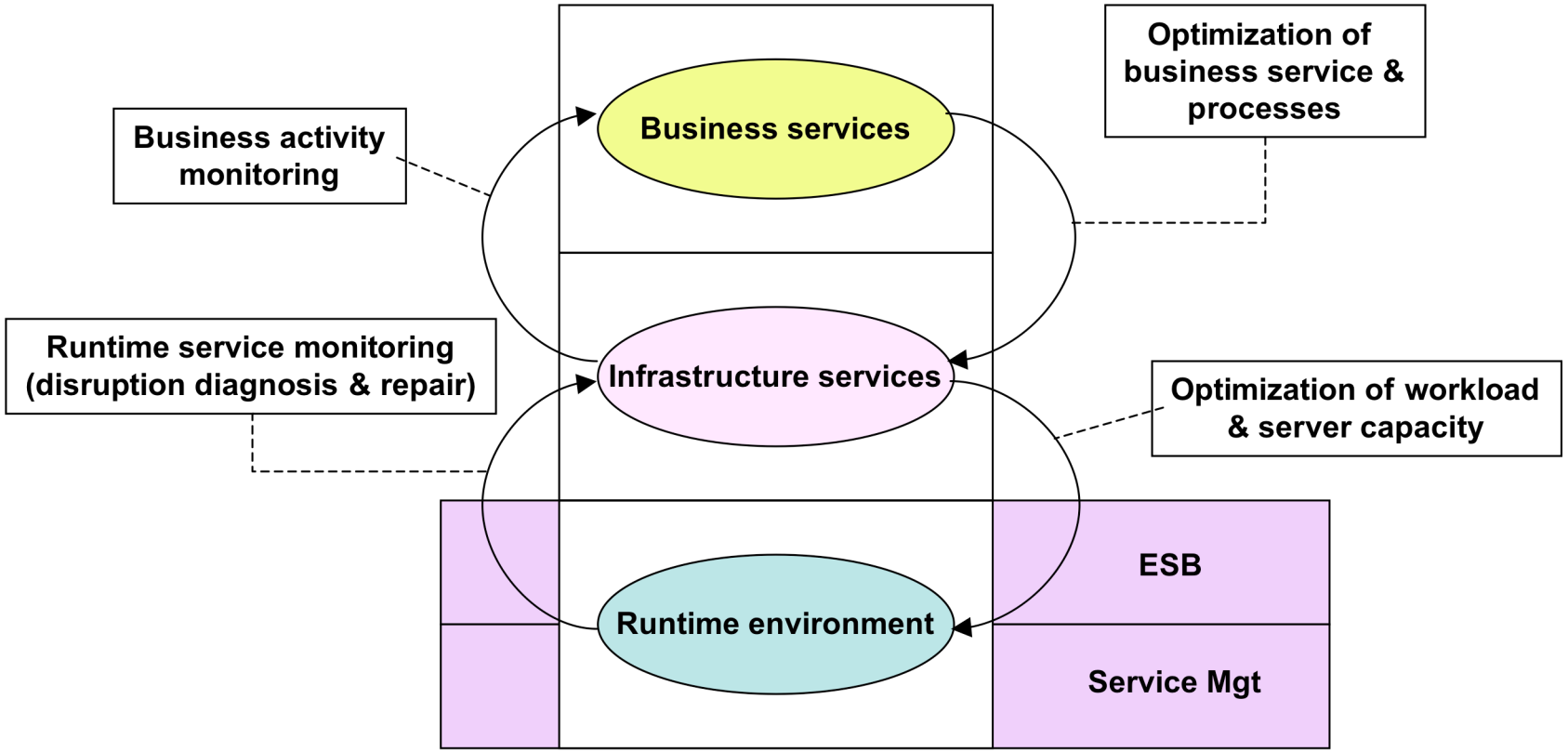


- WS Distributed Mgt is a mgt protocol of info & capabilities in a distributed services environment. WSDM focuses:
  - Management Using WS (MUWS) uses Web services technologies as the foundation of a modern distributed systems mgt. It defines how to describe manageability capabilities of resources using WSDL documents.
  - Management of WS (MOWS) addresses the specific requirements for managing Web services themselves just like any other resource.

4. SOA Governance

- Services that flow between enterprises have defined owners with established ownership and governance responsibilities, including gathering requirements, design, development, deployment, and operations management for *any mission critical or revenue generating service*.
- SOA governance refers to the organization, process, policies and metrics that are required to manage an SOA successfully.
- SOA governance is a formalization of the structured relationships, procedures and policies that ensure the IT architecture in an organization supports and is aligned to business functions, with a specific focus on the life cycle of services. SOA governance is primarily designed to:
  - enable enterprises to maximize business benefits of SOA such as increased process flexibility, improved responsiveness, and reduced IT maintenance costs.
- Two different governance models are possible: central governance versus federated governance.

SOA Governance



For more info. Refer to videotaped Overview and Panel Session: “SOA Governance & the Enablement of Transformation” Robert Johnson IBM & Dave Chappell Oracle (moderators) ICSOC’07 (icsoc.org)

Agenda

- *Introduction*
- *The Multiple Facets of a Service*
- *Service Oriented Architecture*
- *Summary & Pointers to Open Research Problems*

Summary & Pointers to Open Research Problems

- SOAs empower enterprises with a flexible infrastructure & processing environment by provisioning independent, reusable automated business processes (as services) & providing a robust foundation for leveraging these services.
- Effective SOAs must rely on a set of core of core enablers: engineering and planning the SOA, SOA implementation, SOA management and monitoring, and SOA governance.

Open Research Problems

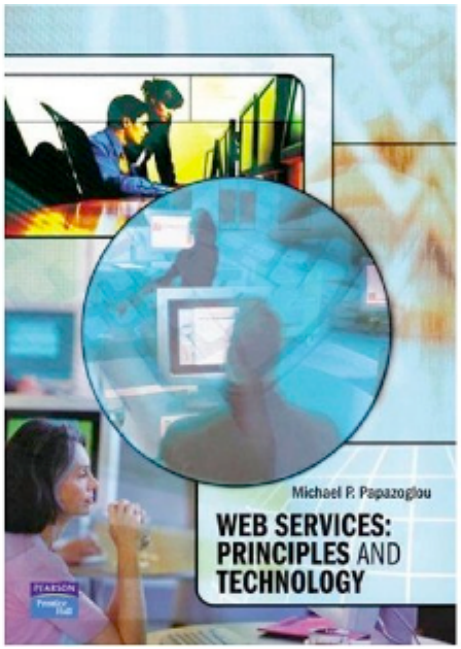
Architectural support is required for:

- Dynamically (re-)configurable run-time architectures:
  - The run-time service infrastructure should be able to configure itself and be optimized automatically in accordance with specific application requirements, QoS, compliance requirements & high-level policies.
- Autonomic Management of Processes:
  - Self- adapting & configuring, self- protecting & healing processes.

Related Research Material

BOOKS

- M. P. Papazoglou “Web Services: Principles and Technology”, *Prentice Hall*, 852 pages, Aug. 2007.
- M. P. Papazoglou, P.M.A. Ribbers “e-Business: Organizational and Technical Foundations”, *J. Wiley & Sons*, 722 pages, March 2006.
- W. J. van den Heuvel “Aligning Modern Business Processes and Legacy Systems”, *MIT Press*, 208 pages, Feb. 2007.
- D. Georgakopoulos, M.P. Papazoglou “Readings in Service Oriented Computing”, *MIT Press*, 658 pages, March 2008.



RESEARCH PAPERS:

- M. P. Papazoglou, W. J. van den Heuvel Service-Oriented Design and Development Methodology, *Int. J. of Web Engineering and Technology (IJWET),* vol. 2, no. 4, 2006, pp. 412-442.
- M. P. Papazoglou, W.J. van den Heuvel "Service Oriented Architectures: Approaches,Technologies and Research Issues", *VLDB Journal*, vol. 16, July 2007, pp. 389-415.
- M. P. Papazoglou, W.J. van den Heuvel “Business process development life cycle methodology”, *Communications of the ACM*, vol. 50 , no.10, Oct. 2007, pp.79 - 85.
- M. P. Papazoglou, P. Traverso, S. Dustdar, F. Leymann “Service-Oriented Computing: State of the Art and Research Directions”, *IEEE Computer*, Oct. 2007.