Modulo III
Service Oriented Architecture

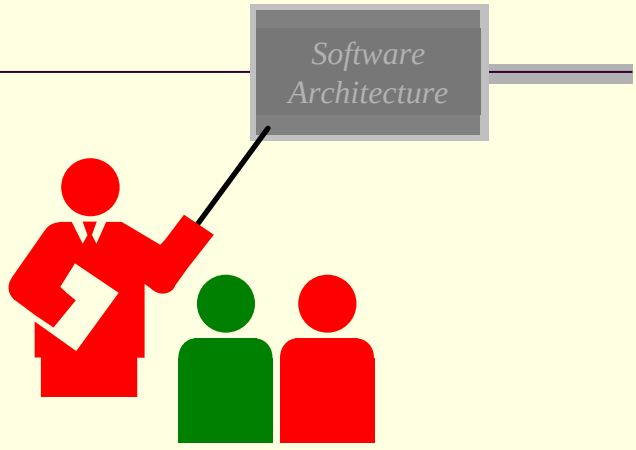
Prof. Ismael H F Santos

Bibliografia

Ementa

- Software Architecture
- SOA
- Evolution

SOA



What is Software Architecture ?

- IEEE 1471-2000
  - Software architecture is the **fundamental organization** of a system, embodied in its **components**, their **relationships** to each other and the environment, and the **principles** governing its design and evolution
- Other definitions of Software Architecture
  - collection of the fundamental decisions about a software product/solution designed to meet the project‘s quality attributes. Includes the main components, their main attributes, and their collaboration expressed in several levels of abstraction (depending on the project's size).
  - Architecture is the first design artifact where a system’s quality attributes are addressed

Why Architecture?

- Architecture serves as the blueprint for the system but also for the project:
  - Team structure
  - Documentation organization
  - Work breakdown structure
  - Scheduling, planning, budgeting
  - Unit testing, integration
- Architecture establishes the communication and coordination mechanisms among components

What is a Service ?

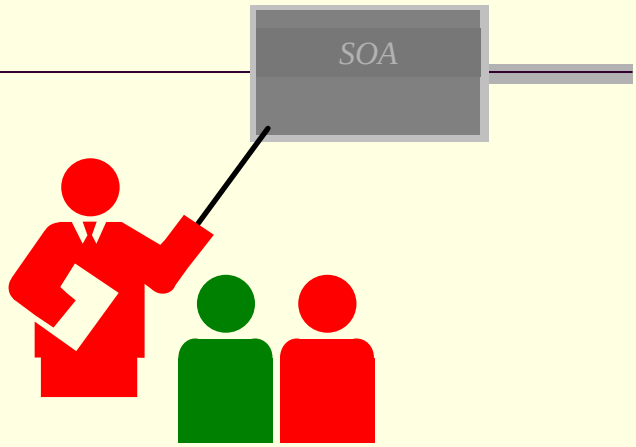
- A facility supplying some public demand the work performed by one that serves HELP, USE, BENEFIT
- In economics and marketing, a service is the non-material equivalent of a good. Service provision has been defined as an economic activity that does not result in ownership, and this is what differentiates it from providing physical goods.
- It is claimed to be a process that creates benefits by facilitating either a change in customers, a change in their physical possessions, or a change in their intangible assets.

 en.wikipedia.org/wiki/Service

What is a service (2)

- A Windows Service?
  - RPC Locator, EventLog, DHCP Client,
- Software Service?
  - Distribution Service, Alert Service
  - Security Service, Log Service
- Business Service?
  - Common Operational Picture, Navigation
  - Accounts Receivable, Customers

SOA



SOA

- **Service-oriented architecture** other than its literal translation that it is an architecture that relies on service-orientation as its fundamental design principle. Service-orientation describes an architecture that uses loosely coupled services to support the requirements of business processes and users

- **Architecture** is not tied to a specific technology. It may be implemented using a wide range of technologies, including REST, RPC, DCOM, CORBA or Web Services.

What is SOA ?

- *What is SOA?*
  - *SOA is an architectural style whose goal is to achieve loose coupling among interacting software agents. A service is a unit of work done by a service provider to achieve desired end results for a service consumer. Both provider and consumer are roles played by software agents on behalf of their owners. [xml.com]*
  - *Characteristics*
    - Documented and discoverable interfaces (APIs)
    - Standards-Based
    - Infrastructure neutral – use of H/W, OS, and S/W used to implement the service is encapsulated from the service consumer
- *What is a "Web Service"?*
  - *A Web service is a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by internet protocols [w3.org]*
- *"Web-Enabled" does NOT EQUAL Web Service nor SOA*

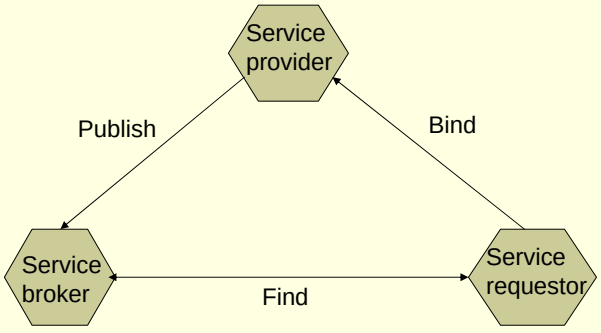
SOA Definitions

- **SOA** is a design for linking business and computational resources (principally organizations, applications and data) on demand to achieve the desired results for service consumers (which can be end users or other services). OASIS (the Organization for the Advancement of Structured Information Standards) defines SOA as :
  - *A paradigm for organizing and utilizing distributed capabilities that may be under the 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.*

Origin of Service-Oriented Architecture

- IBM has created a model to depict Web services interactions which is referred to as a "service-oriented architecture" comprising relationships among three entities (see next slide):
  - A Web service provider;
  - A Web service requestor; and a
  - A Web service broker.
- Note: IBM's service-oriented architecture is a generic model describing service collaboration, not specific to Web services.
  - See http://www-106.ibm.com/developerworks/webservices/

Origin of Service-Oriented Architecture



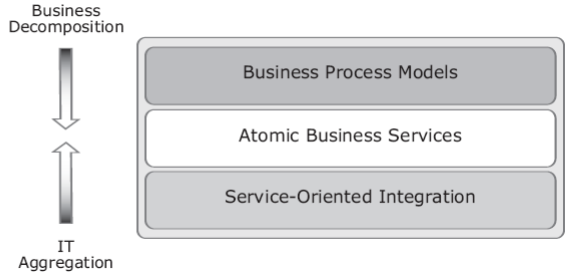
Service-oriented architecture representation (Courtesy of IBM Corporation)

SOA bridges the gap between the business and IT.

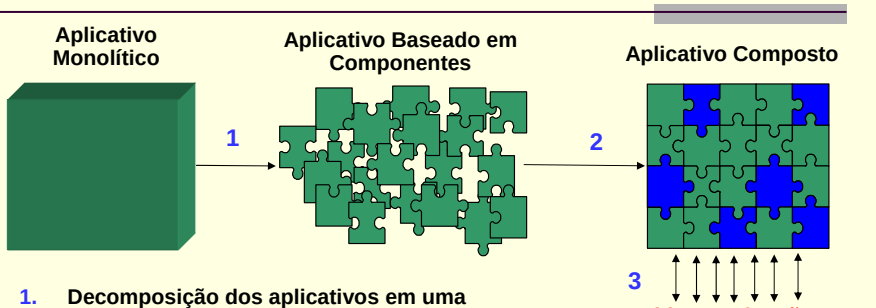
- The business disaggregates its business process models downward toward the atomic business services layer.
- IT aggregates software assets upward toward that layer using service-oriented integration.
- They meet in the middle and negotiate a consensus view of the necessary atomic business services.

Figure 1

Converging on atomic business services.

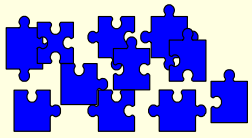


SOA

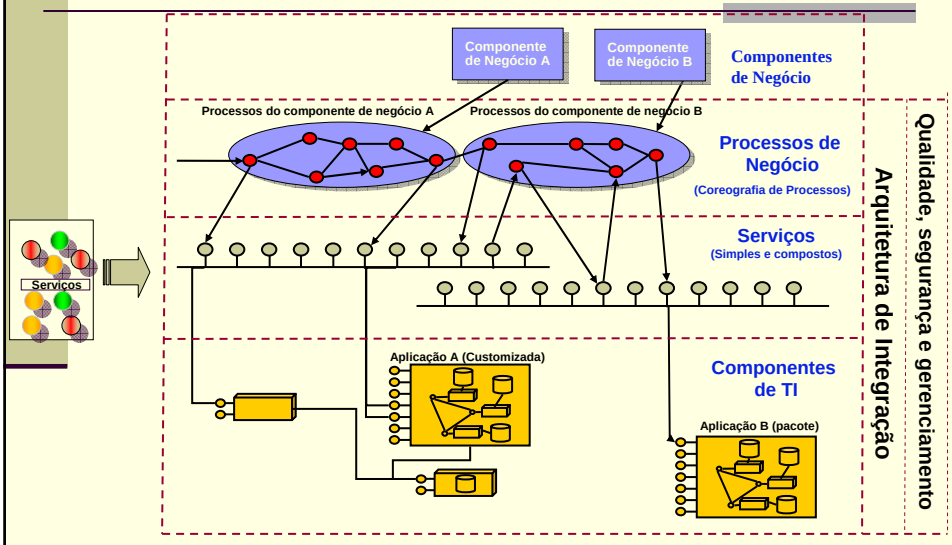


1. **Decomposição dos aplicativos em uma representação modular e reutilizável baseada em funções e serviços**
2. **Combinar as funções e serviços para atender as demandas da Empresa**
3. **A reutilização dos serviços para atender novas oportunidades ou ameaças de negócios.**

Recombinar as funções e os serviços para atender novos requisitos de negócios



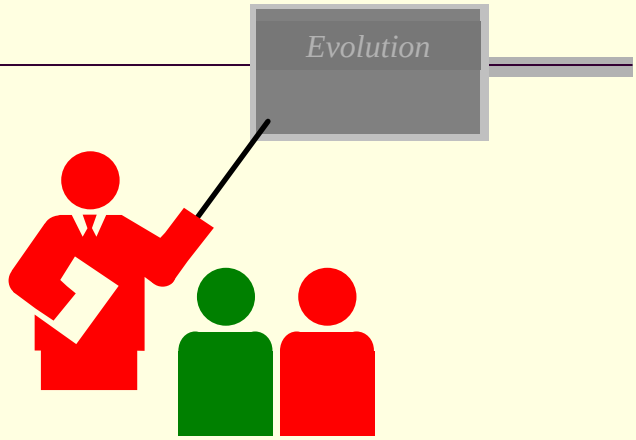
SOA Vision



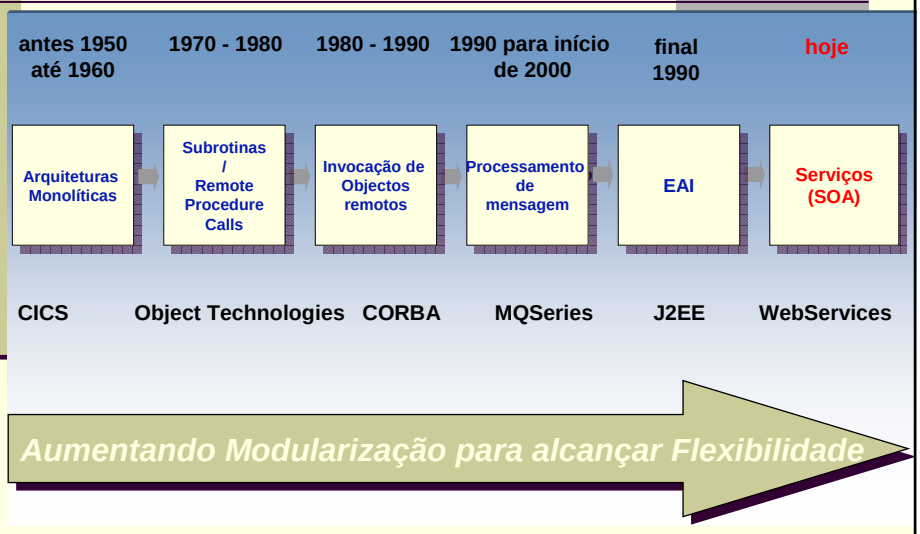
Why SOA ?

- The main drivers for SOA adoption are that it links computational resources and promotes their reuse. **Enterprise architects** believe that SOA can help businesses respond more quickly and cost-effectively to changing market conditions.
- This style of architecture promotes reuse at the macro (service) level rather than micro level (objects). It can also simplify interconnection to - and usage of - existing IT (legacy) assets.
- SOA promotes the goal of separating users (**consumers**) from the service implementations. **Services** can therefore be run on various distributed platforms and be accessed across networks. This can also maximize reuse of services

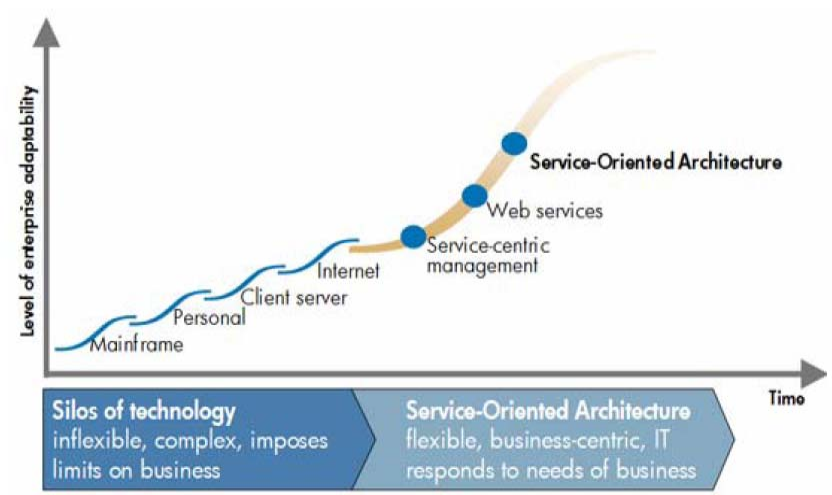
SOA



SOA: evolution to a more flexible TI architecture



SOA: evolution to a more flexible architecture



SOA: flexibility on current investments

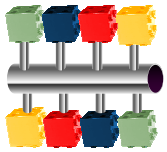
Backbone de Mensageria

- Conexão Ponto-a-Ponto entre aplicações
- Conexão simples e básica

Enterprise Application Integration (EAI)

- EAI conecta aplicações via um hub centralizado
- Facilita o gerenciamento de um grande número de conexões

Service Orientated Integration

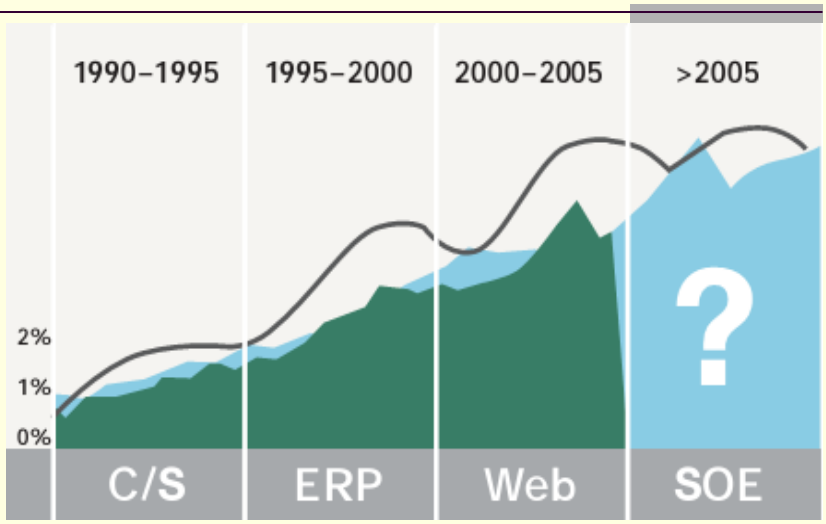


- Integração e coreografia de serviços através do Enterprise Service Bus
- Conexões flexíveis com interfaces bem definidas e baseadas em padrões

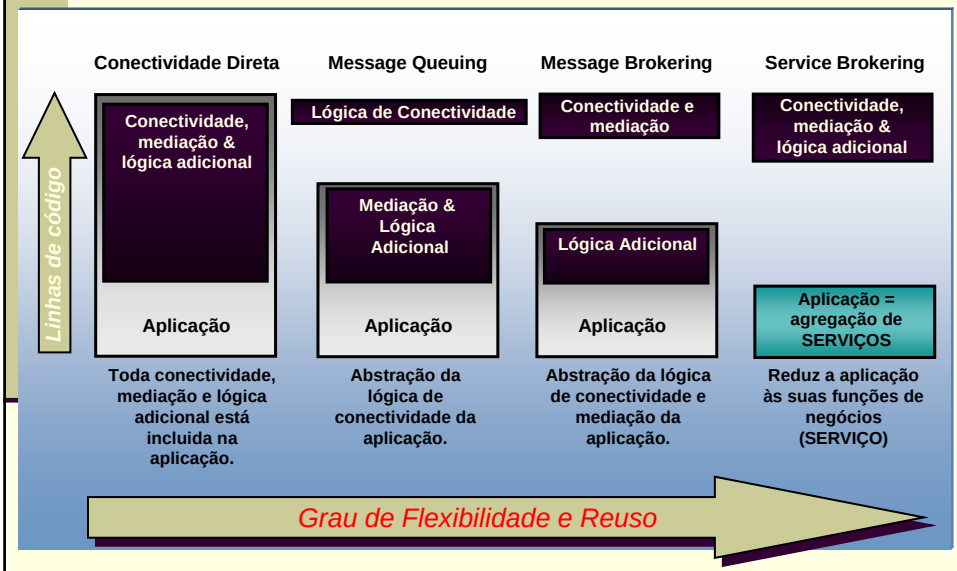
Flexibildade

Evolução dos padrões e infraestrutura de middleware

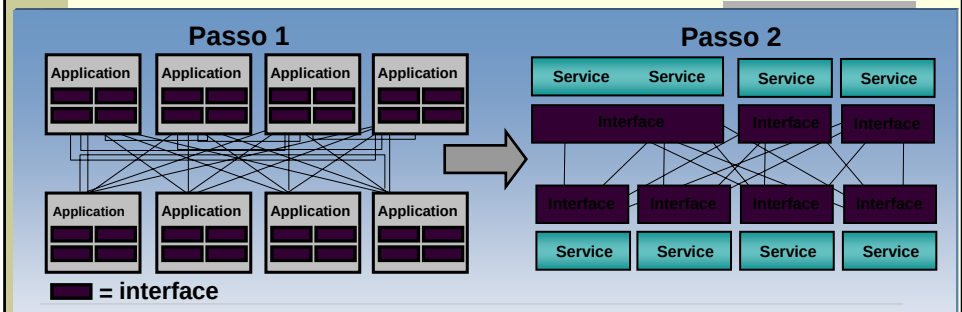
Towards Service Oriented Enterprise



SOA: next connectivity step



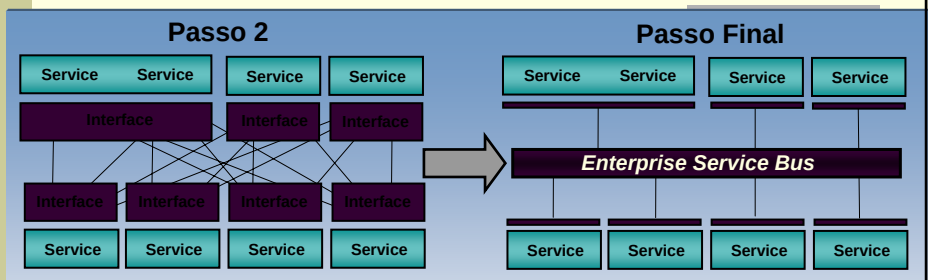
Enterprise Service Bus



- ✓ A descrição das interfaces é voltada para negócios
- ✓ Retira as interfaces das aplicações de negócios
- ✓ Reduz o número e a complexidade técnica das interfaces
- ✓ Permite a reutilização das aplicações e das interfaces de negócios

Connection still is´point-to-point ….

ESB eliminates ponito-to-pont conection



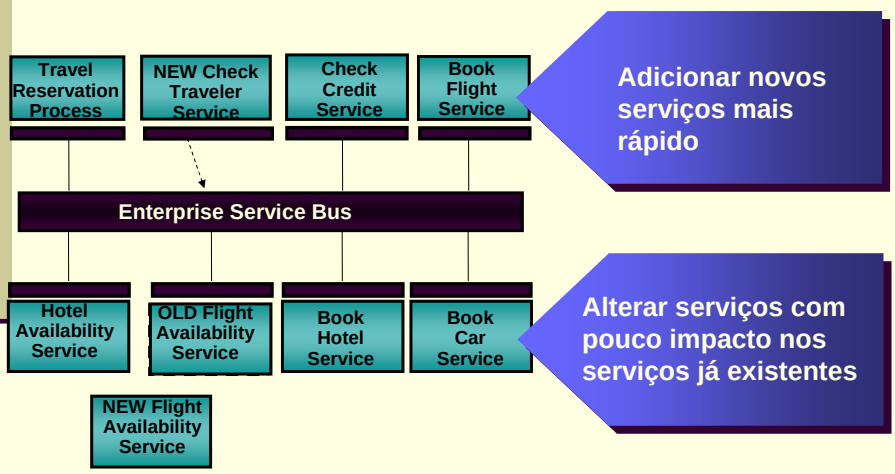
- ✓ Retira as conexões ponto-a-ponto das interfaces
- ✓ Permite um acoplamento mais flexível das aplicações
- ✓ Utilização de um broker para interação entre as interfaces
- ✓ Permite o reuso das interfaces de conexão.

RESULT → greater flexibility & agility

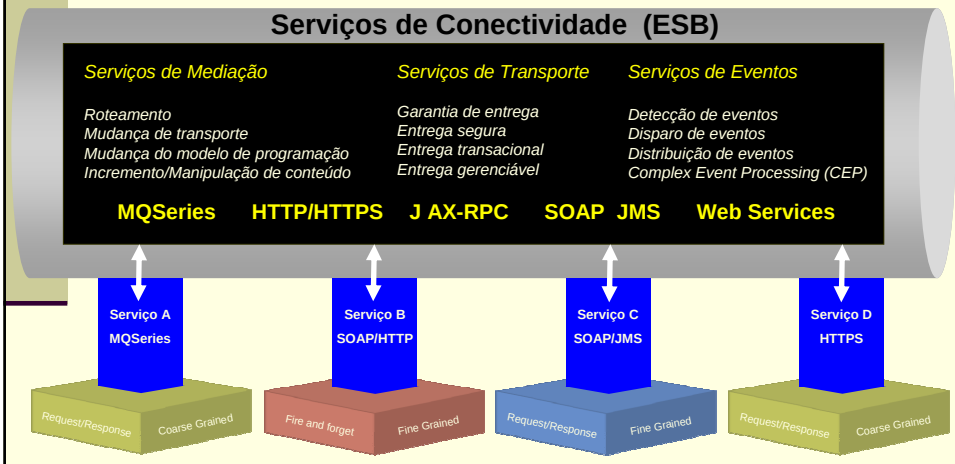
Enterprise Service Bus (ESB)

- ESB provides the mechanism for continuously adapting the portfolio of available services to accommodate shifting needs.
- Rather than requiring constant updates to constituent services, the ESB can adaptively mediate among them.
  - For example, if two services use slightly different formats for their messages, the ESB can translate between them.
  - Alternately, if a business change in one area requires a new version of a service, the ESB can route requests that require enhanced functionality to the new version while routing requests that assume only basic functionality to the old version.

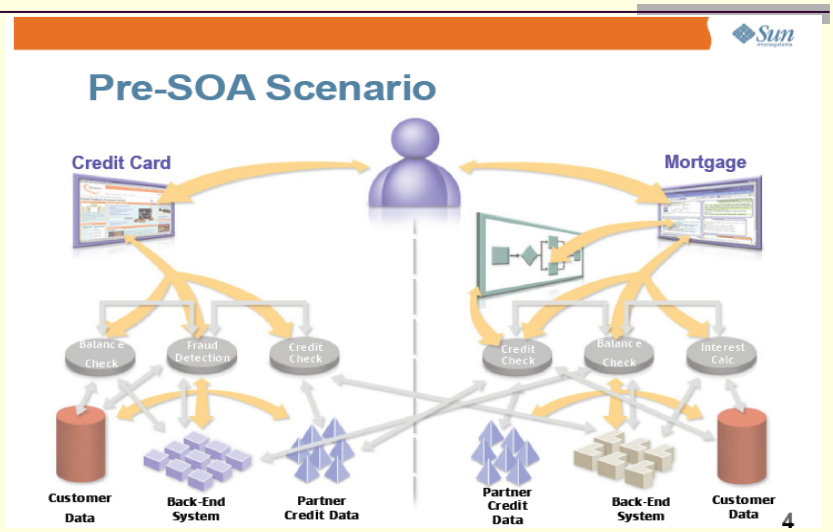
ESB easily accept newcomers



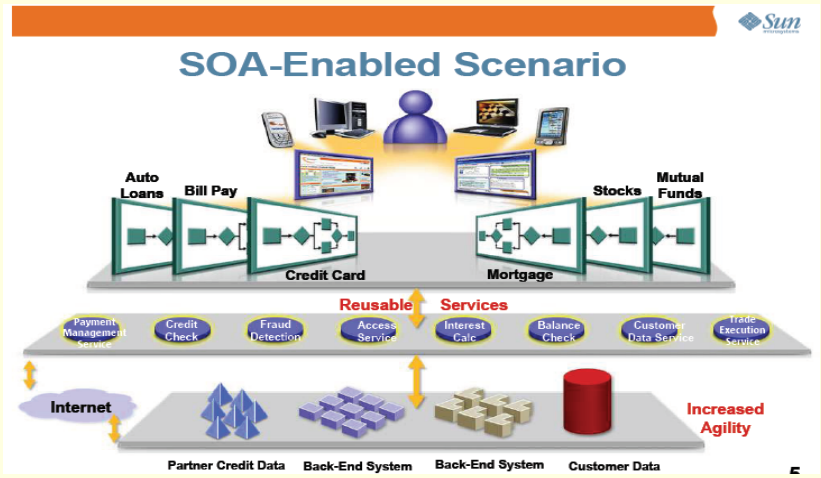
ESB – Separation of concerns



SOA



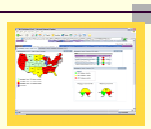
SOA



Mudança para SOA

De	Para
■ Orientado a função ■ Construído para durar ■ Longos ciclos de desenvolvimento	■ Orientado a processo ■ Construído para mudanças ■ Construído e implementado incrementalmente
■ Silos de aplicação ■ Forte acoplamento ■ Estruturação das aplicações usando componentes e objetos ■ Implementação conhecida	■ Soluções orquestradas ■ Fraco acoplamento ■ Estruturação das aplicações usando serviços ■ Abstração de implementação

O que é uma aplicação ?



User Interaction Logic

Data Logic

Integration Logic

Process Logic

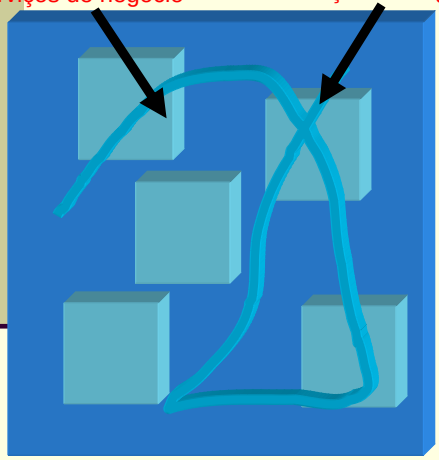
Business Rules

Infrastructure Logic

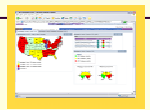
Serviços de negócio

Serviços de negócio

Função de negócio



Integração através da desintegração



User Interaction Logic

Data Logic

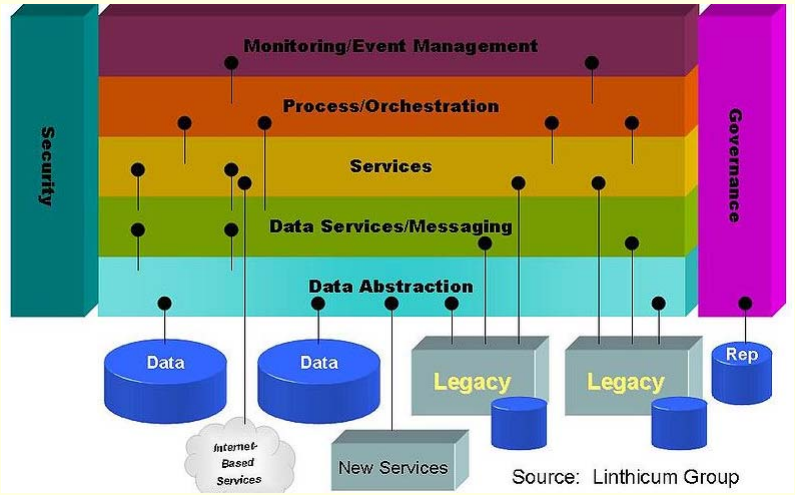
Integration Logic

Process Logic

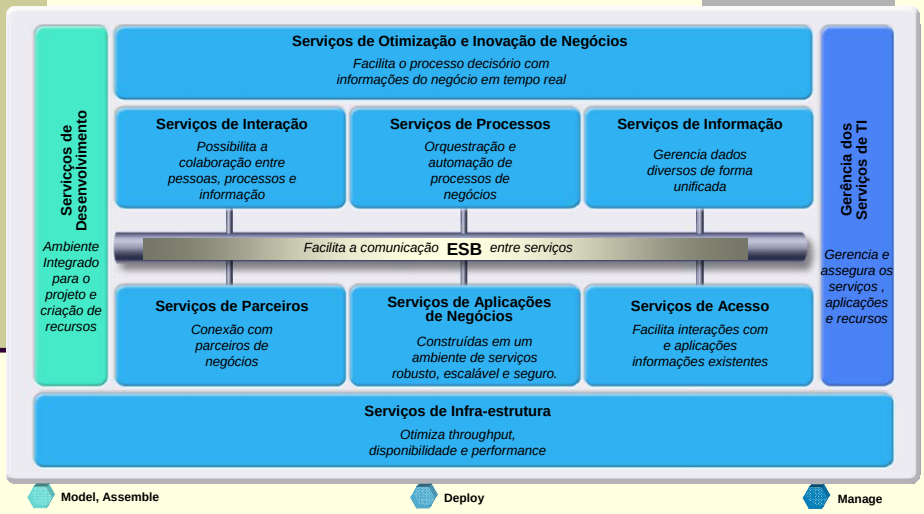
Business Rules

Infrastructure Logic

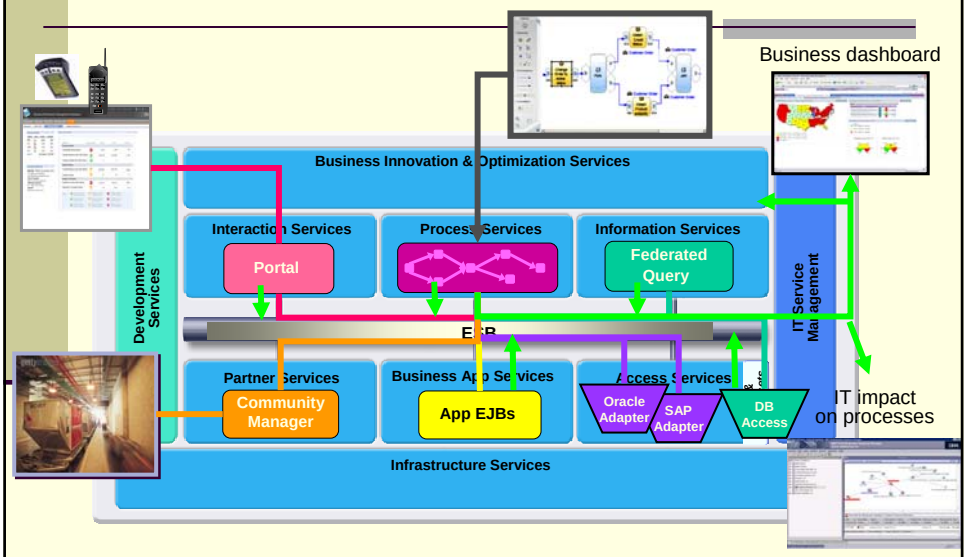
SOA Meta-Model



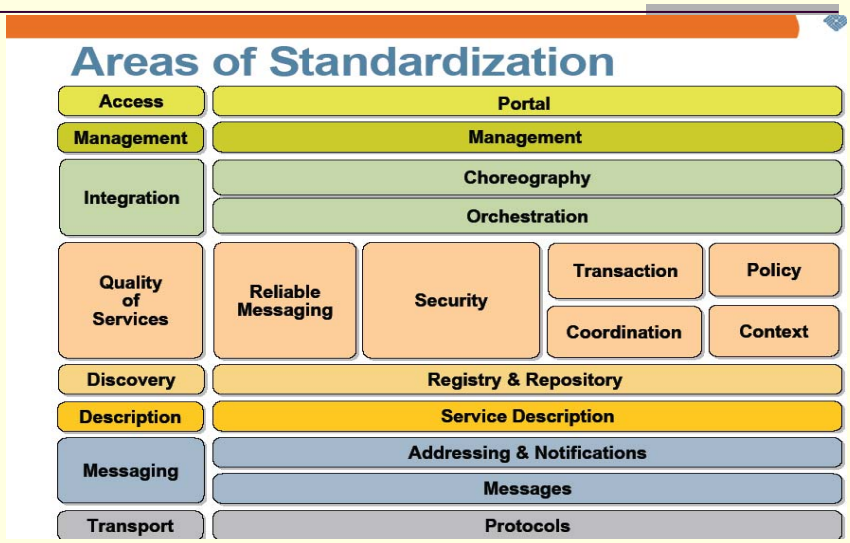
Arquitetura de Referência SOA – Separation of concerns



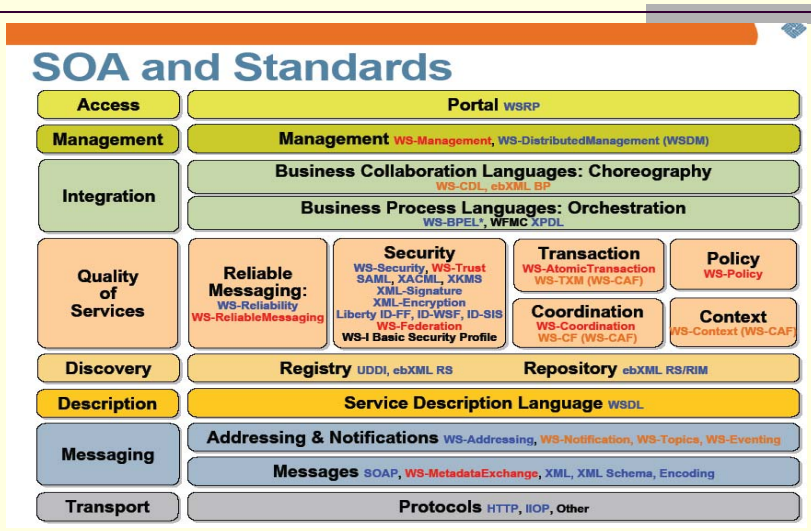
SOA Echology



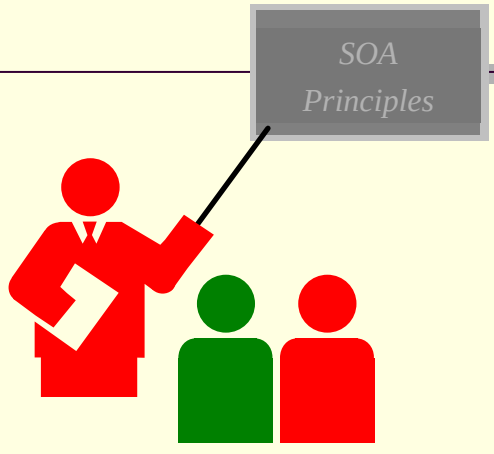
SOA Reference Model



SOA Reference Model



SOA



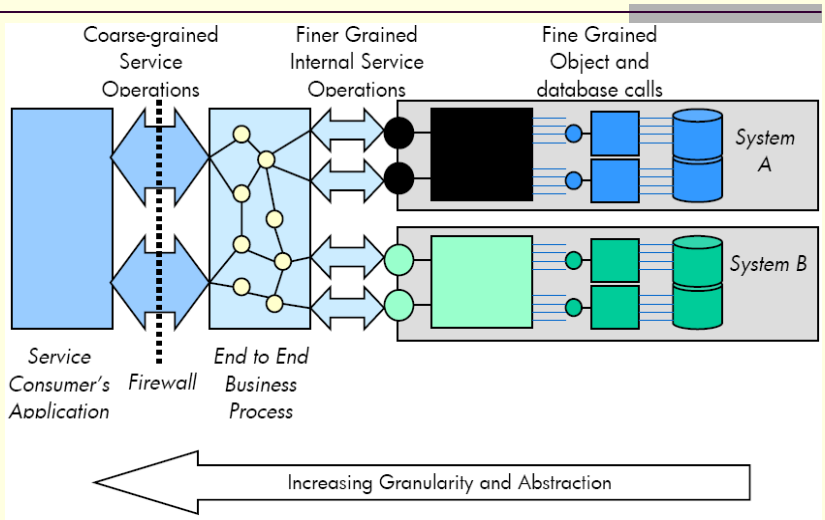
SOA Architectural Principles (1)

- **Service Encapsulation**
- **Service Loose coupling** - Services maintain a relationship that minimizes dependencies and only requires that they maintain an awareness of each other
- **Service contract** - Services adhere to a communications agreement, as defined collectively by one or more service description documents
- **Service abstraction** - Beyond what is described in the service contract, services hide logic from the outside world
- **Service documentation** - A description of a service-oriented design must contain at least three separate uses of the phrase "business value".

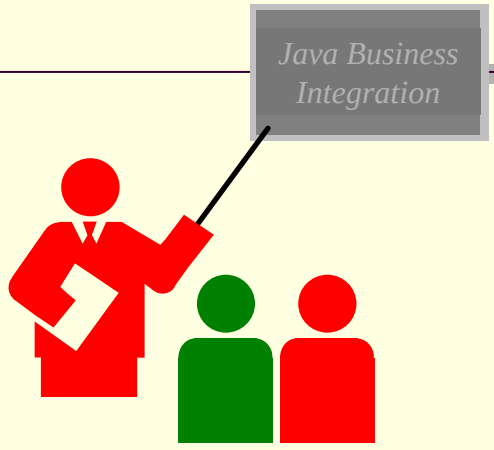
SOA Architectural Principles (2)

- **Service reusability** - Logic is divided into services with the intention of promoting reuse
- **Service composability** - Collections of services can be coordinated and assembled to form composite services
- **Service autonomy** – Services have control over the logic they encapsulate
- **Service optimization** – All else equal, high-quality services are generally considered preferable to low-quality ones
- **Service discoverability** – Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms

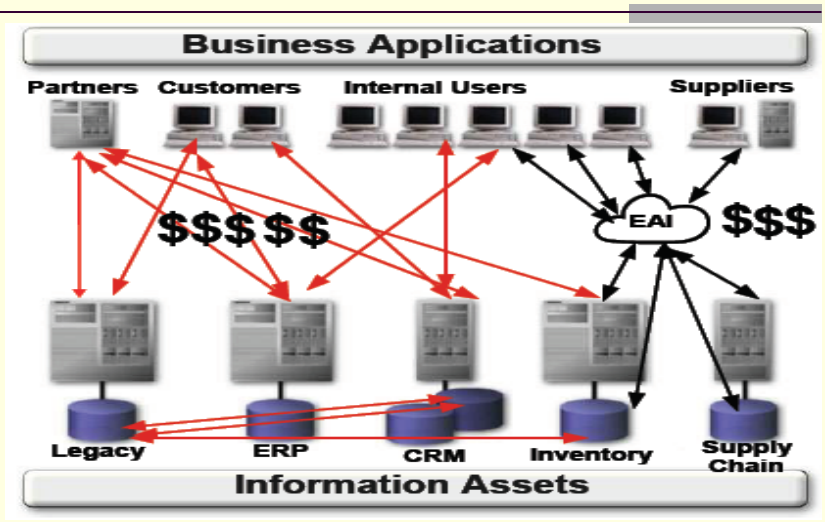
Service Granularity



SOA



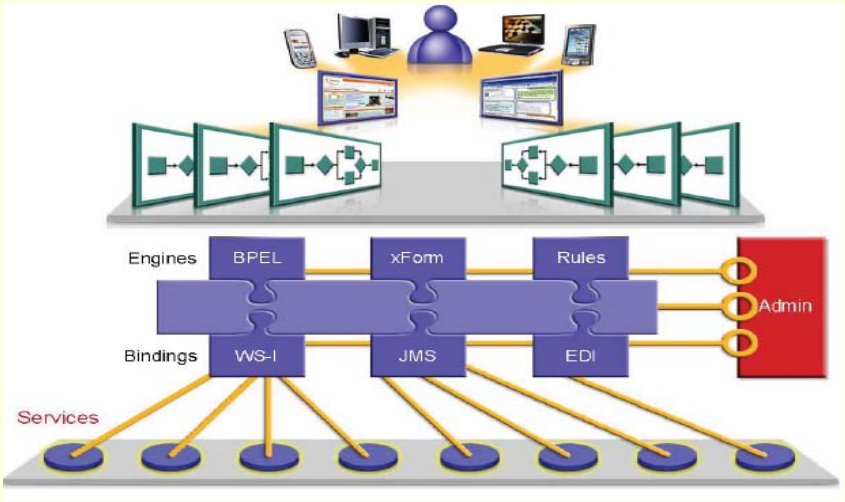
Pre-JBI Based Integration Approaches



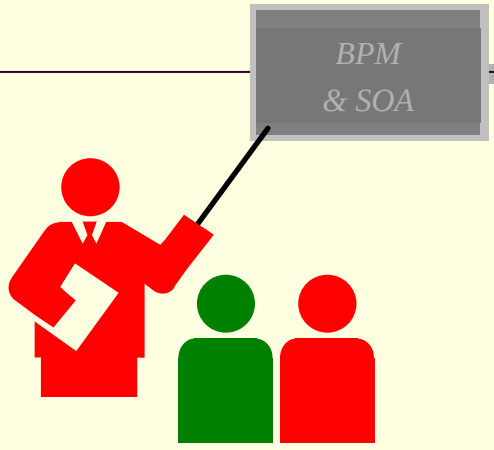
What is JBI (JSR-208)?

- One of the biggest motivation for SOA is to reduce the cost of application integration
- Standard-based, pluggable infrastructure
  - Service Engines: Provides business logic and integration services
  - Binding Components: Provides communications protocols
- JBI does for application integration what J2EE did for application development

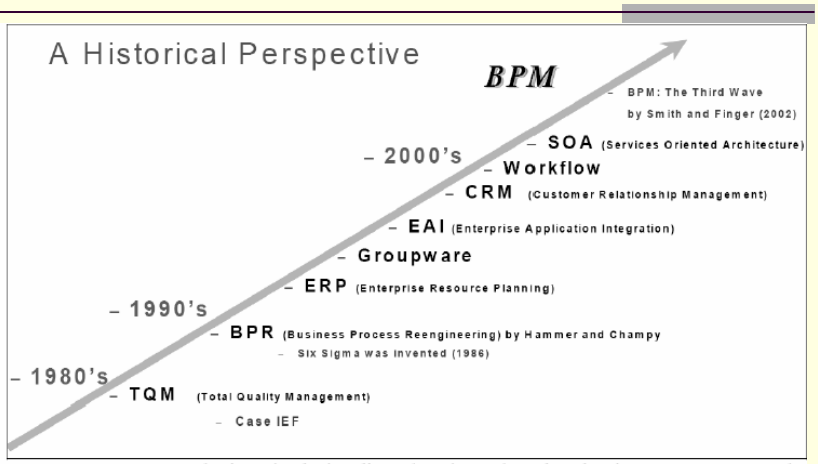
JBI Architecture



SOA

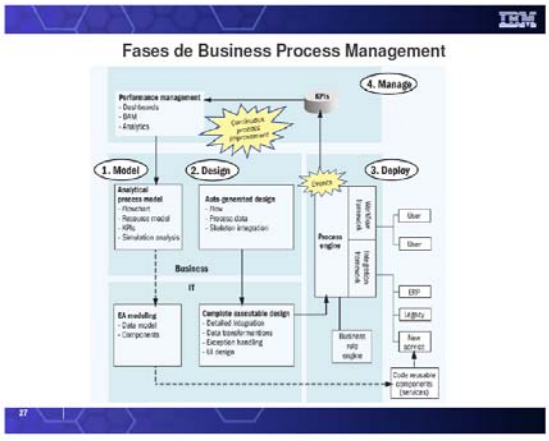


BPM



General Historical Timeline showing when key business concepts and technologies became widely recognized.

BPM



BPM

BPM: Not Your Father's BPR

What's Really New?	Implications
• **Process orientation complements** (not replaces) functional orientation	BPM transforms the organization
• Processes must be **effective and transparent**, not just efficient	Process must be explicit, not embedded in apps
• Continuous improvement is not sufficient; **incremental improvement** must be harmonized with transformative change	Iterate; don't just re-engineer Design for change, not to last
• Business transactions must be more responsive to customer and market demands; thus, **processes must be adjustable**, not perfect	Internal & external process participants can change process

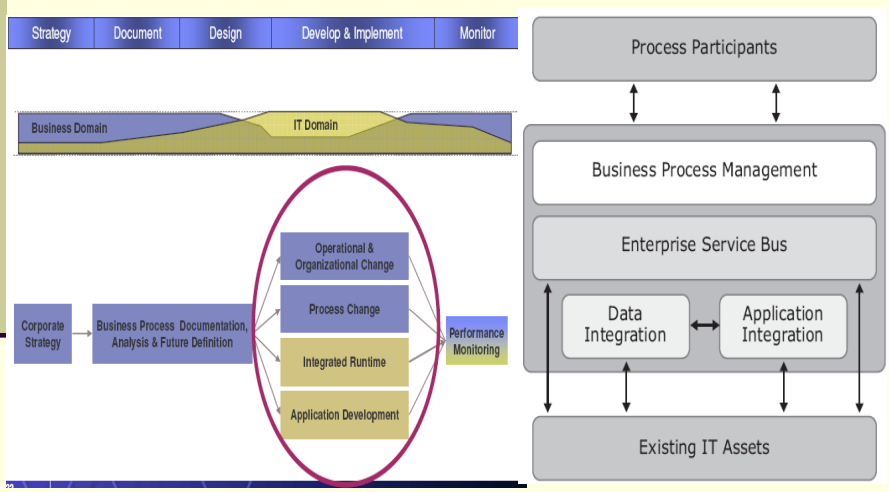
Gartner

BPM

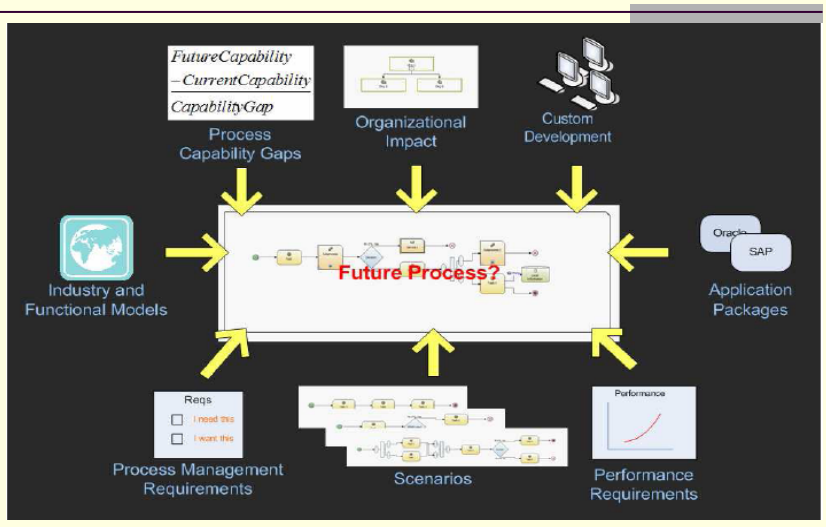
Suggested Business Process Management: Foundational Domains

Technology Infrastructure	Related Services	Organizational Impacts	Expertise & Experience	Process Activities
▪ Architecture ▪ Web Services ▪ Standards ▪ Platforms ▪ Portals ▪ Process Engines ▪ EAI ▪ Frameworks ▪ Templates	▪ Consulting – Strategy ▪ Process – Technology ▪ Implementation – Design – Build – Train ▪ Business Process Outsourcing	▪ Culture ▪ Organizational Structure ▪ Roles/ Responsibilities ▪ Policies and Incentives ▪ Governance/ Compliance ▪ Change Management ▪ Quality Improvement ▪ Best Practices	▪ Skills ▪ Training Courses ▪ Education ▪ Curriculum ▪ Certification ▪ Research ▪ Institutions: – Universities – Colleges – Trade Schools ▪ Government at all Levels	▪ Discovery ▪ Modeling ▪ Monitoring ▪ Deployment ▪ Execution ▪ Analysis ▪ Simulation ▪ Repositories ▪ Activities – Functional – Business – Architecture

BPM


Strategy
Document
Design
Develop & Implement
Monitor
Business Domain
IT Domain
Corporate Strategy
Business Process Documentation, Analysis & Future Definition
Operational & Organizational Change
Process Change
Integrated Runtime
Application Development
Performance Monitoring
Process Participants
Business Process Management
Enterprise Service Bus
Data Integration
Application Integration
Existing IT Assets

BPM


FutureCapability
– CurrentCapability
CapabilityGap
Process Capability Gaps
Organizational Impact
Custom Development
Industry and Functional Models
Future Process?
Oracle
SAP
Application Packages
Reqs
Process Management Requirements
Scenarios
Performance
Performance Requirements

BPM & SOA

Gartner Research

Publication Date: 9 February 2007 ID Number: G00145586

SOA and BPM Are Better Together

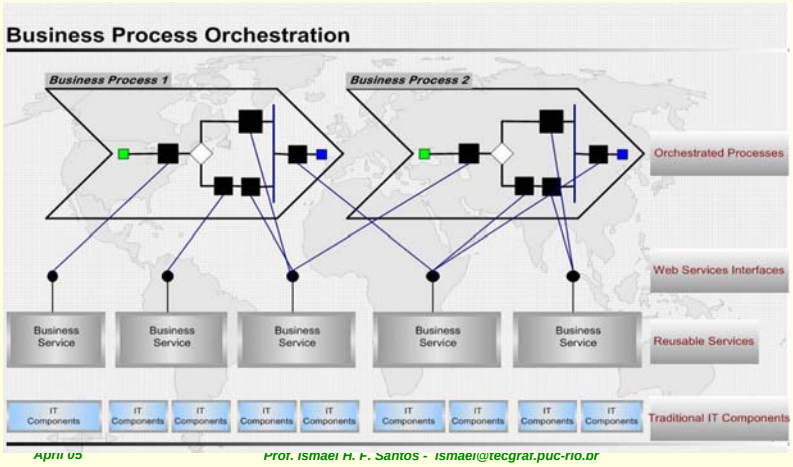
Paolo Malinverno, Janelle B. Hill

Although many user organizations have separate service-oriented architecture and business process management initiatives under way, both are more successful and the benefits are compounded when they are united.

BPM refers to the newest process management discipline in which business processes are viewed as assets to be managed, designed and continuously improved to enhance business agility and operational performance. BPM is a structured approach employing methods, policies, metrics, management practices and software tools to manage and continuously optimize an organization's activities and processes. BPM-enabling technologies make a process explicit — that is, visible and readily changed. BPM-enabling technologies separate a process model from its implementation. The process model is independent of technologies (including applications, data and infrastructure) that may be used in its deployment. The newest, most complete and integrated set of BPM-enabling technologies is defined by Gartner as a business process management suite (BPMS) — see "Business Process Management Suites Enhance the Control and Management of Business Processes" and "Selection Criteria Details for Business Process Management Suites, 2006."

BPM -> SOA infrastrucuture

- BPM provides process modeling, execution, and management components.



Layers of SOA Model

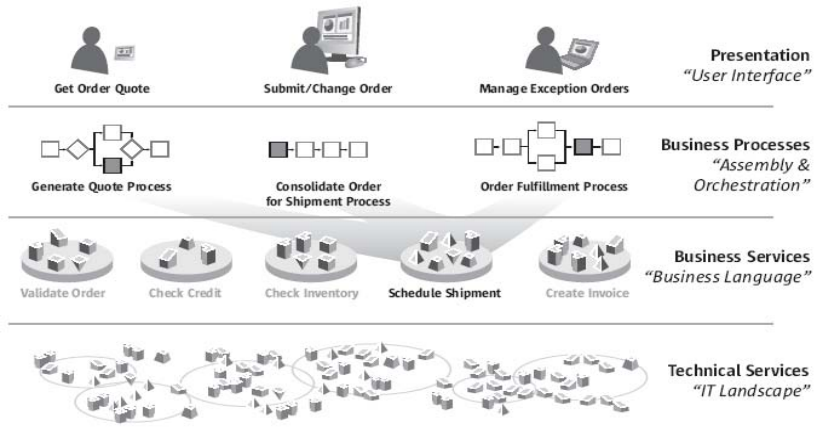


Figure 4: Four-layer SOA model

OMG - Layers of SOA Model

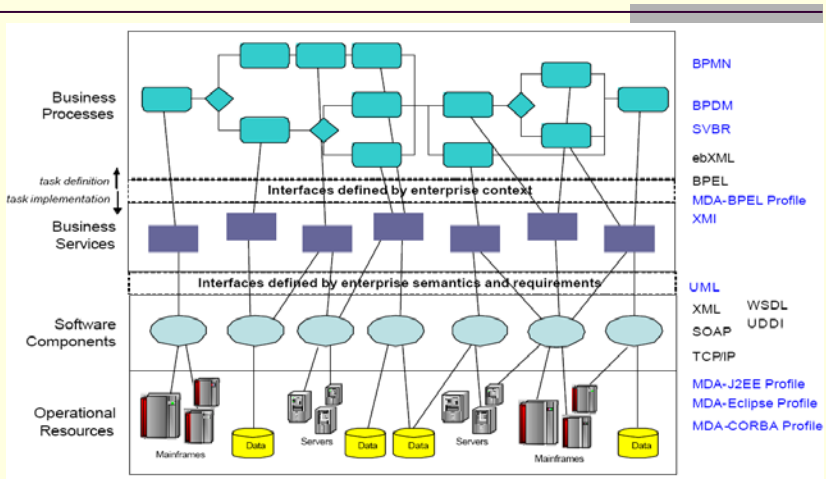
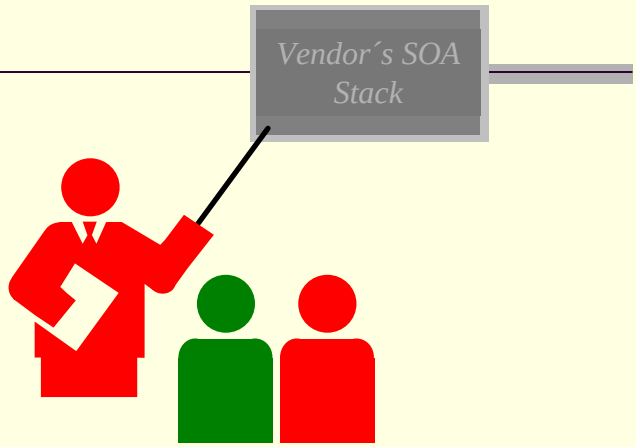


Figure 1. A Service Oriented Architecture. [1]
Some of the OMG MDA Standards that Support SOA are listed on the right in blue.

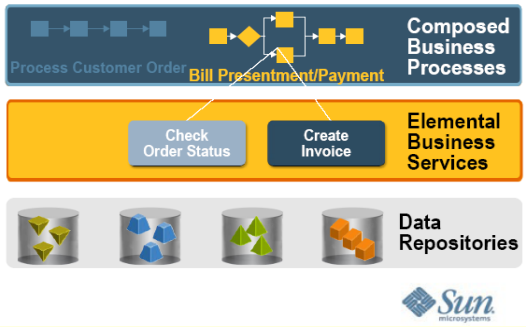
SOA



Sun´s Pragmatic SOA

Automate Business Processes End to End with Composite Applications

- Utilize business analysts for **"process-oriented"** development
- Align IT with **business goals**
- Facilitate **re-usability**
- Adopt **layered** approach to development
- Faster application development, lower **TCO**



Sun´s Pragmatic SOA

Additional Capabilities	Management	Portals	
Business Process Orchestration	Composition/Orchestration		
Composable Service Elements	WS-Security	Reliable Messaging	Transactionality
Messaging	Endpoint Identification, Publish/Subscribe		
Description	XML Schema, WSDL, UDDI, SOAP with Attachments		
Invocation	XML, SOAP		
Transports	HTTP, HTTPS, Others		

Sun Java Composite Application Platform Suite

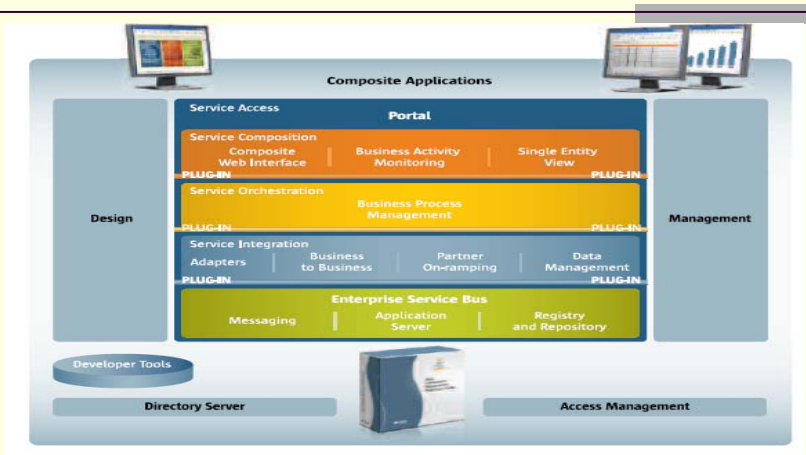


Figure 2. Java CAPS is a fully integrated software suite for building an S composite applications.

Sun Java Composite Application Platform Suite

- **Service access** – A layer of presentation services that provide transparent access to multiple device types and simplify the task of integrating Web services into na enterprise portal environment
- **Service composition** – Tools for composing Web pages with a consistent user interface and a development environment for building reusable software components that are based on Web services standards

Sun Java Composite Application Platform Suite

- **Service orchestration** – Tools for business process management and assembling composite applications that can be used to create new services in support of business processes
- **Service integration** – Legacy or packaged applications can be integrated into the SOA environment using service integration tools and pre-built adapters for commonly used enterprise applications
- **Enterprise service bus** – Core integration services for connectivity, transformation, and routing are provided by the enterprise service bus

Sun´s Pragmatic SOA