Business Consulting Services

Architecture for Integration

Hans-Peter Hoidn

2 October 2003

Agenda

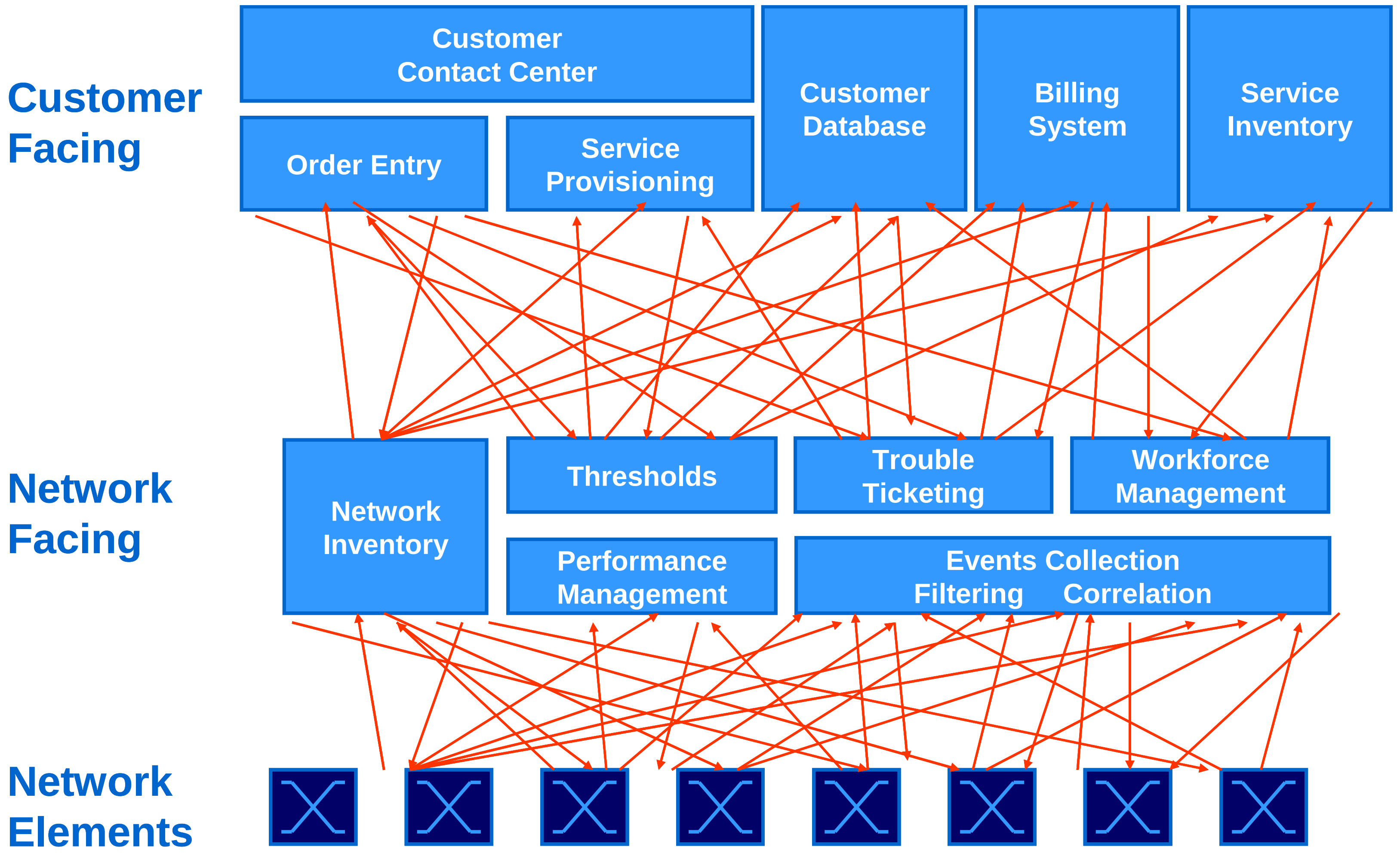
Problem statement

I. Integration Layer in general

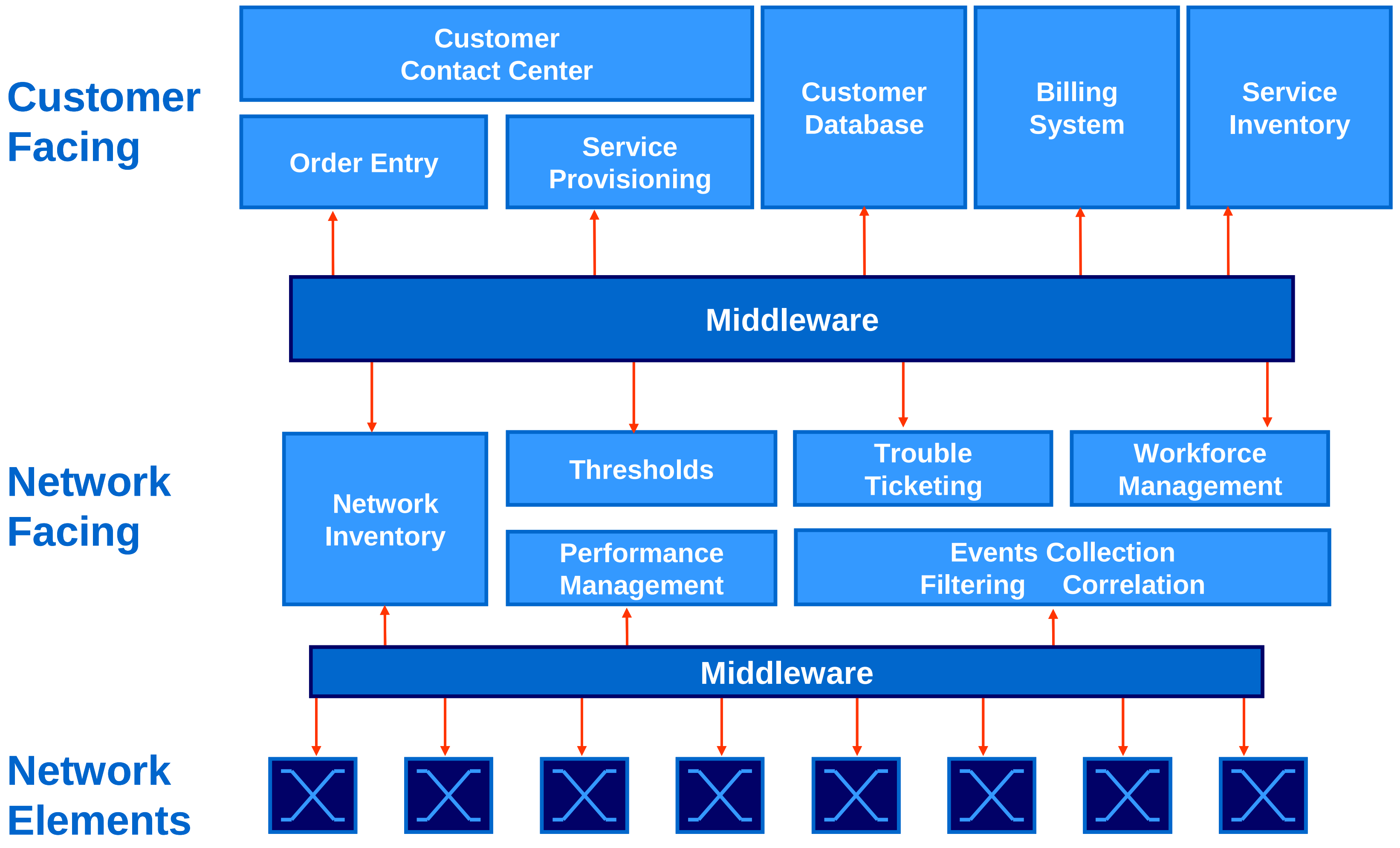
Situation and Approach

- Situation:
  - Successful isolated solutions exist
  - New Challenges
- Objectives:
  - Flexible solutions addressing customer needs
  - „end-to-end“ solutions
  - Re-use of existing applications
- Approach:
  - Integration
  - Coexistence of existing applications and packaged solutions

Business Situation
(Communication Flow among Applications)

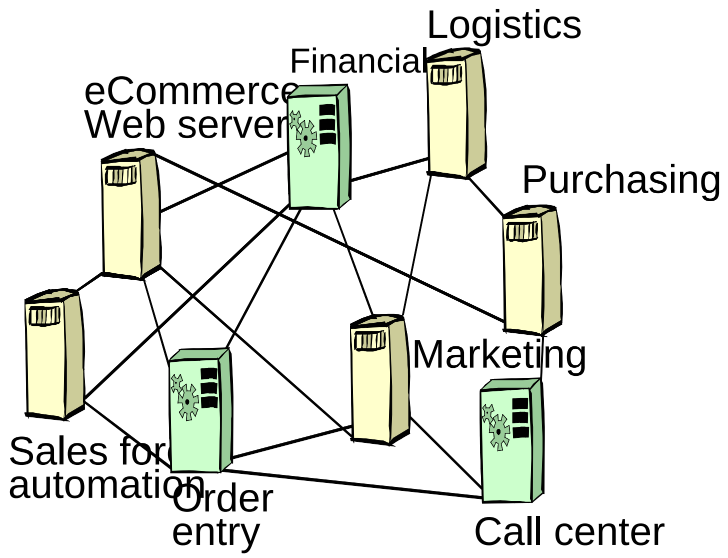


Goal for a Manageable System Architecture



EAI-Problem

Analysts report that more than 80 % of corporate data is exchanged using point-to-point integration.



Given an average of 50 applications, a corporation might need to build 2,450 point-to-point connections to integrate the entire enterprise.

Too complex!

Not manageable!

Not flexible!

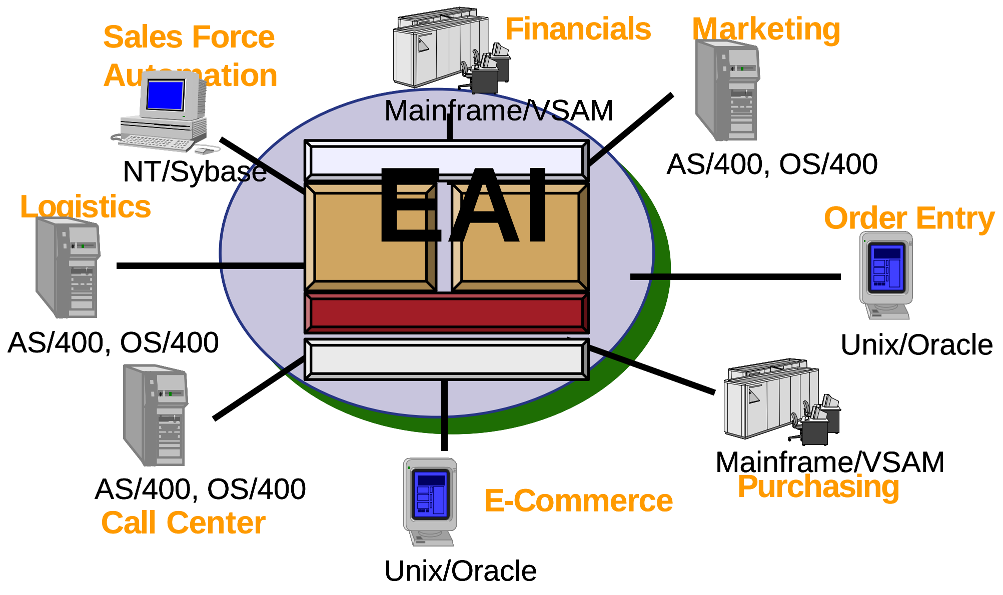
Maintenance costs!

Too many technologies!

Business changes hard to adapt!

EAI Solution

What if there were an "integration layer" that would simplify the tasks of integration and sharing of data between systems?



Integration Layer Characteristics

- Overcome
  - Missing flexibility for new business
  - Historical redundancies
  - Many point to point integrations - critical complexity
- Provide
  - Strong business process orientation
  - Consolidated applications
  - Ease of future changes
  - Smooth legacy replacement
- Addressing
  - Various business systems

Lessons learned - An architectural Approach is Required

- Legacy systems can no longer be adapted to current business needs – need for flexible solutions
- Flexibility cannot be provided by vertical stove pipe solutions which do not support horizontal issues as needed by a customer oriented viewpoint
- Flexible solutions need a component approach providing business components which can be composed to various business systems
- Architectural and integration issues raise in importance –flexible solutions need to be based on solid blueprints for application and technical architecture

II. EAI Environments

II: EAI Environments

- Key Functionality:
  - Asynchronous Messaging
  - Guaranteed Delivery
  - Data Transformation
  - Message Routing
- Components:
  - Core Middleware
  - Adapter and Message Broker
  - Business Process Administration (including Workflow)

EAI Framework – Functional View (Increasing Value of Integration)

- Transportation:
  - Transport from application to application, Routing
- Data Transformation:
  - Mapping of semantics
  - Mapping among protocols (e.g. IDOC to/from XML)
- Business Rules:
  - Business Processes, Data depending Routing
- Business Processes:
  - Business Process Automation, Workflow

EAI Scope

Enterprise Application Integration

Integration spanning the entire enterprise and community.

Not just a few applications.

Communication platform for the applications.

Not just a database replication.

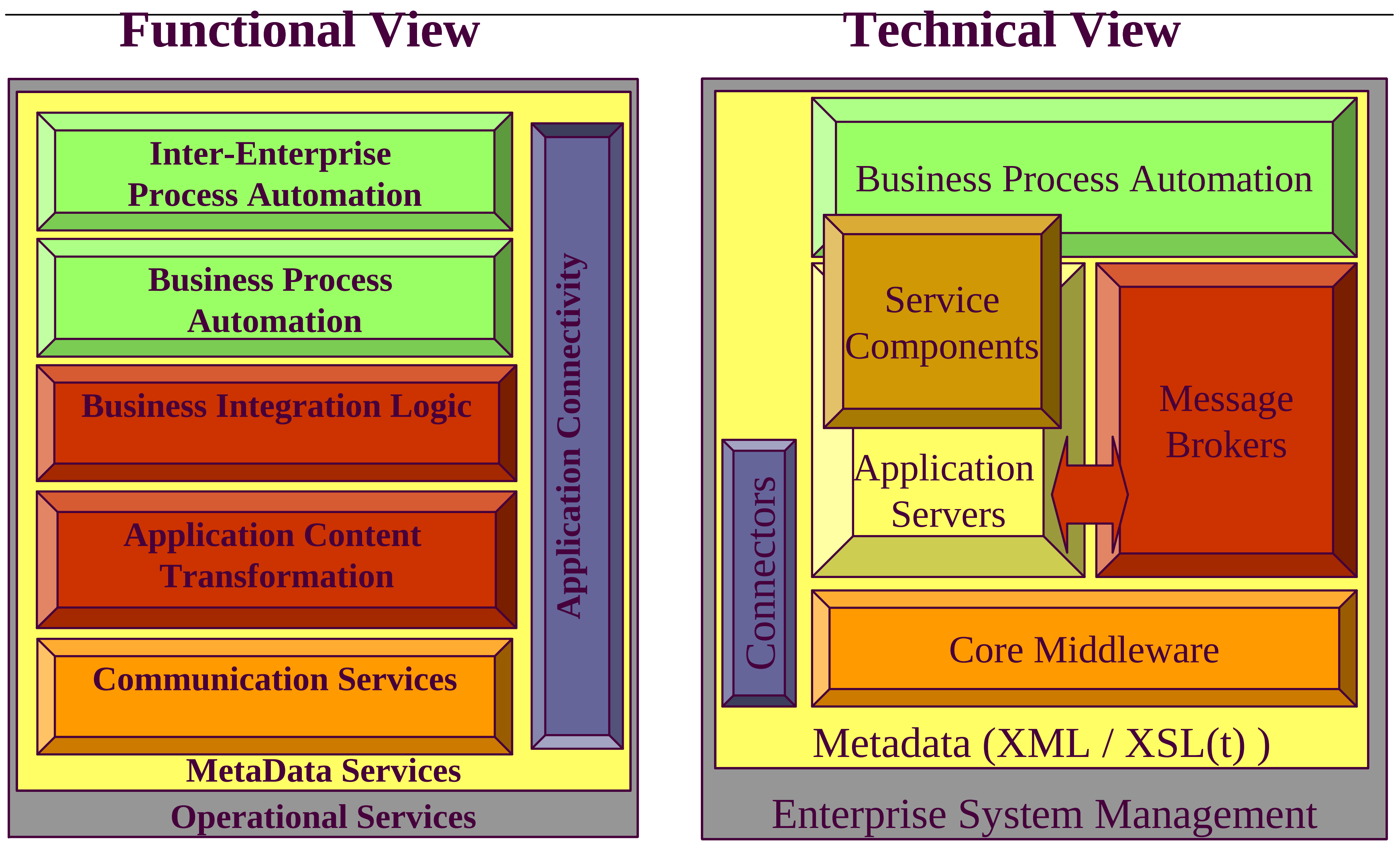
Open system architecture.

Not just an application-to-application interface based interoperability.

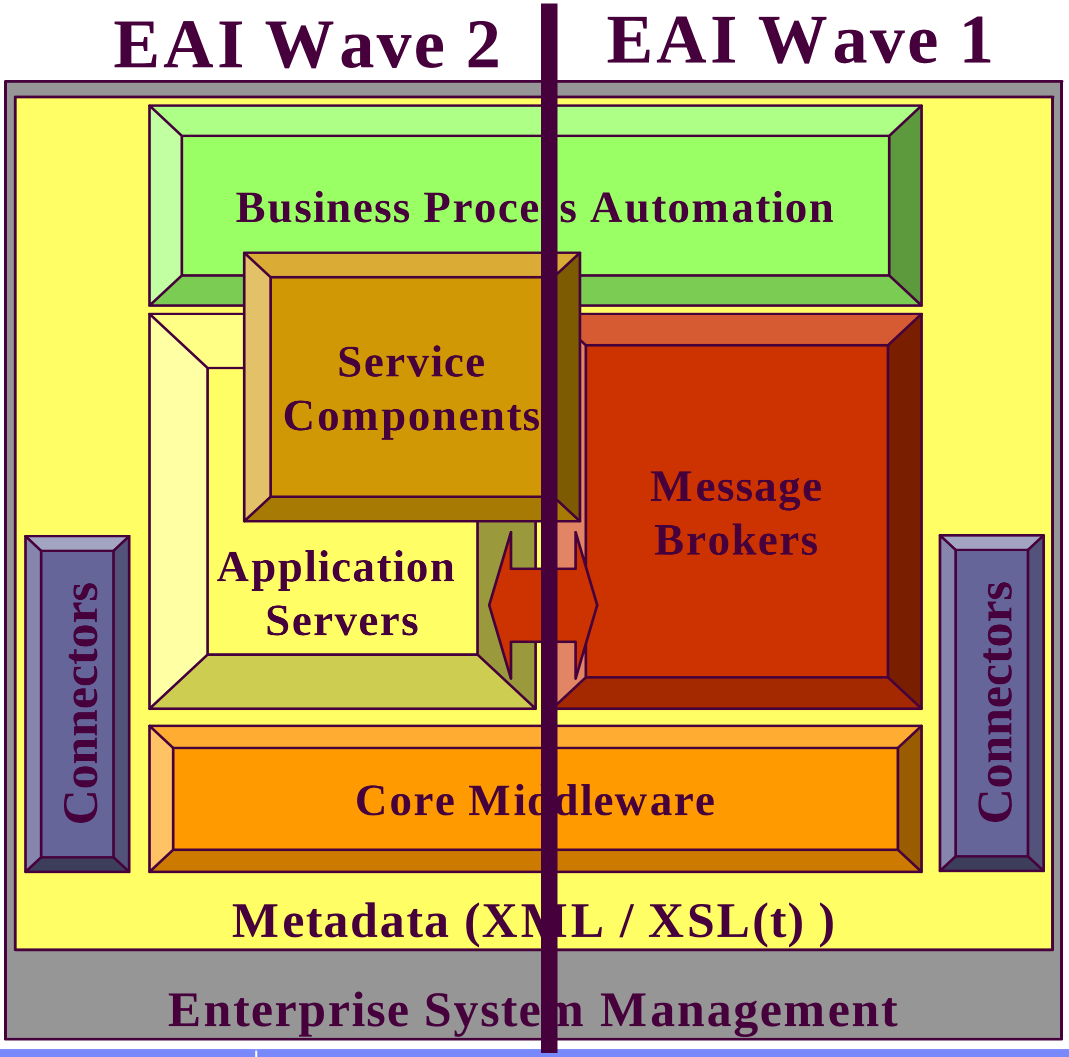
EAI – Key Components

- Core Middleware:
  - Message-Transport (transactional)
- Adaptors / Connectors:
  - Links to applications (e.g. SAP)
- Message Broker:
  - Message Routing and Data Transformation
- Application Server
  - Linking to Program Execution
- Business Process Automation

EAI Functional and Technical Views



EAI Waves

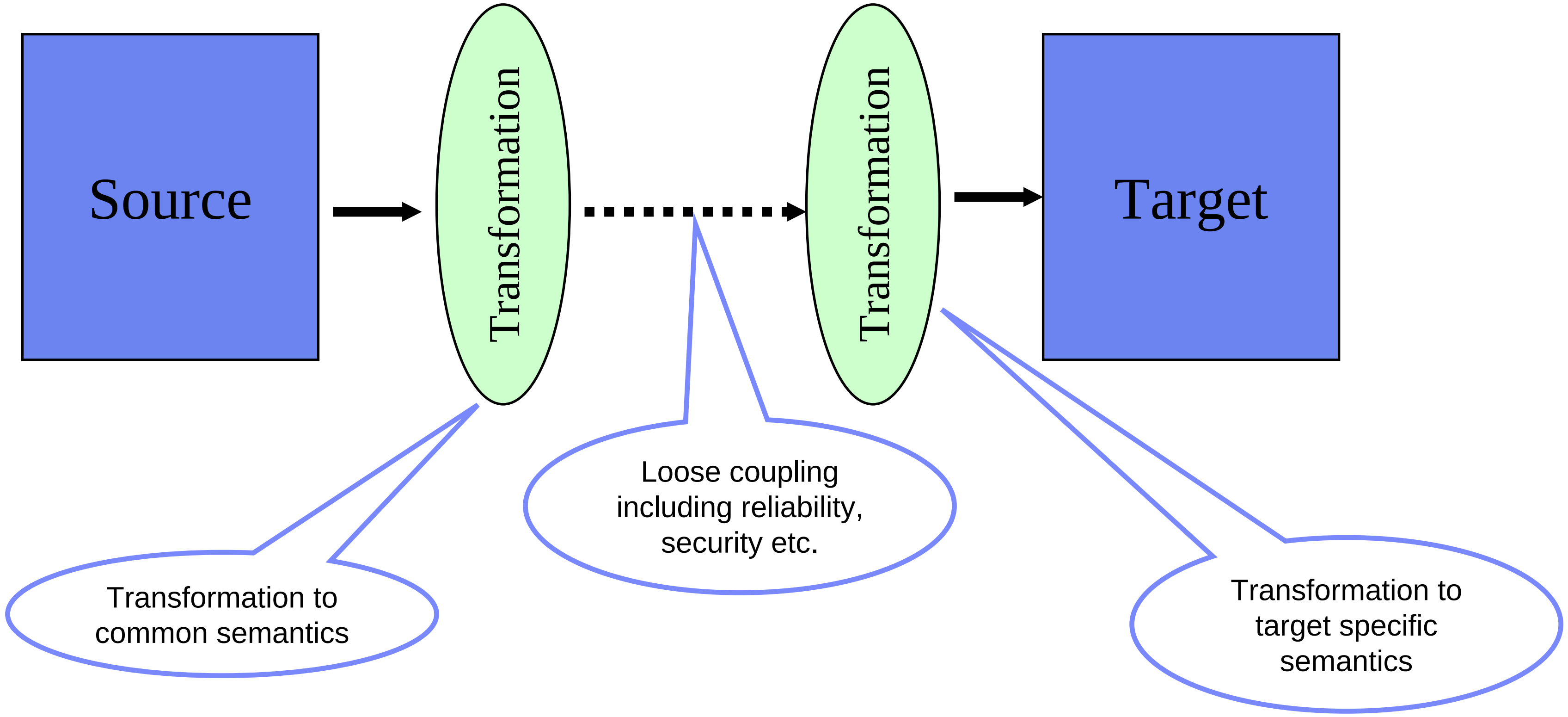


The EAI product landscape is evolving from **packages** which we classify as EAI Wave 1 to **frameworks / platforms** which we classify as EAI Wave2

•EAI Wave1 products are packaged based which you need to configure and extend, based on common approaches but proprietary implementations, and typically utilise proprietary process based run-time environments and connectors

•EAI Wave 2 products are open standards based flexible integration frameworks that benefit from proven scalable, reliable execution run-time environments and are evolving to provide the functionality of EAI Wave 1 products and include EIP functionality.

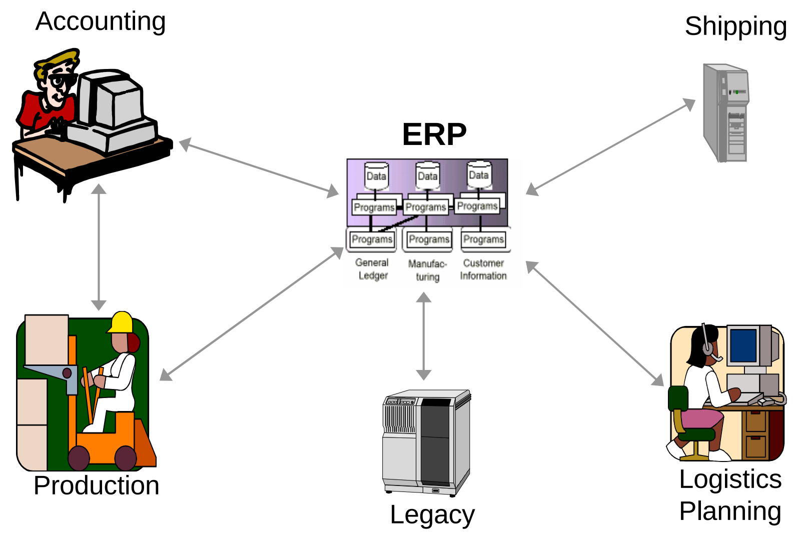
Applications / Components and Information Flow


Source
Transformation
Transformation
Target
Transformation to common semantics
Loose coupling including reliability, security etc.
Transformation to target specific semantics

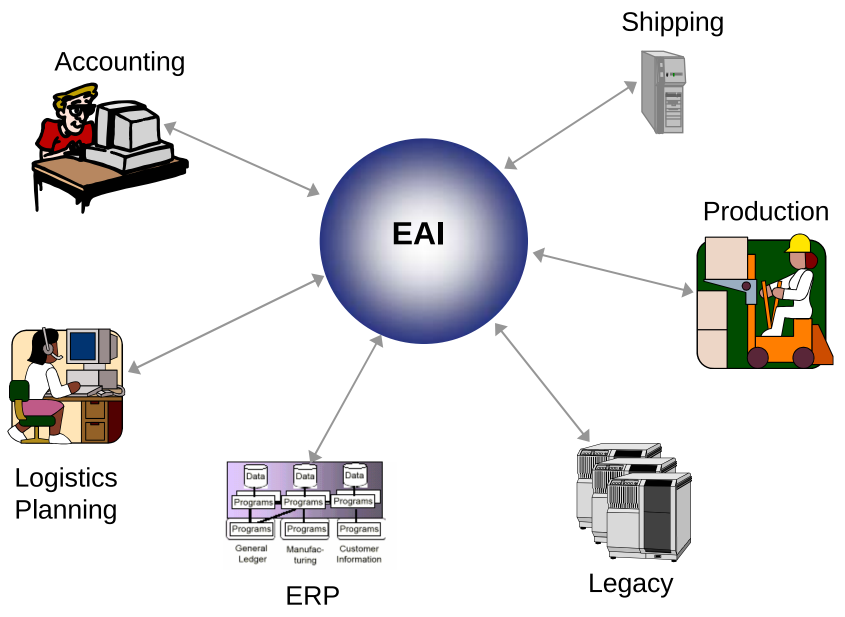
EAI – Example with ERP

Strategic Business Initiatives: Enterprise Resource Planning

Traditional Approach to ERP



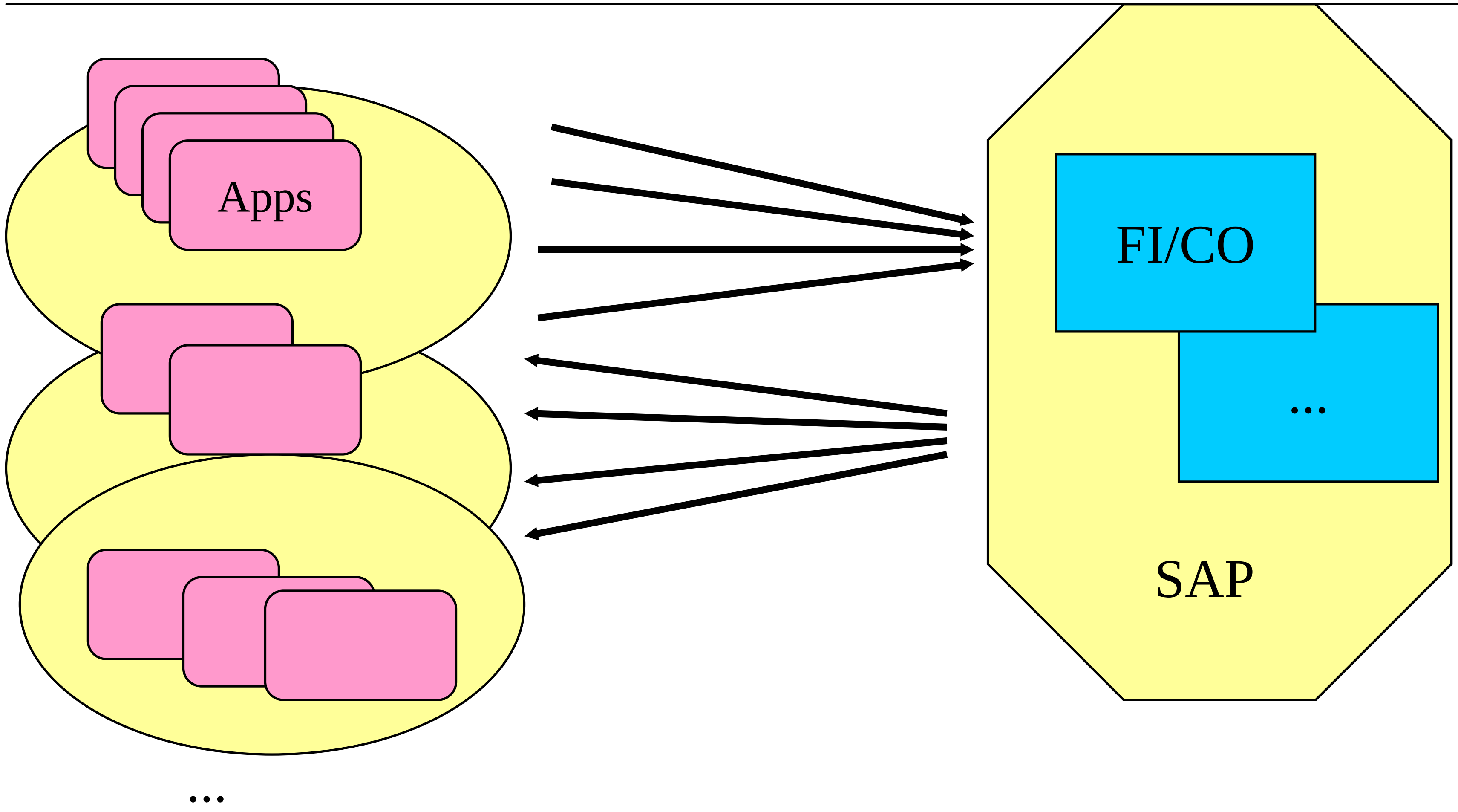
EAI Approach to ERP



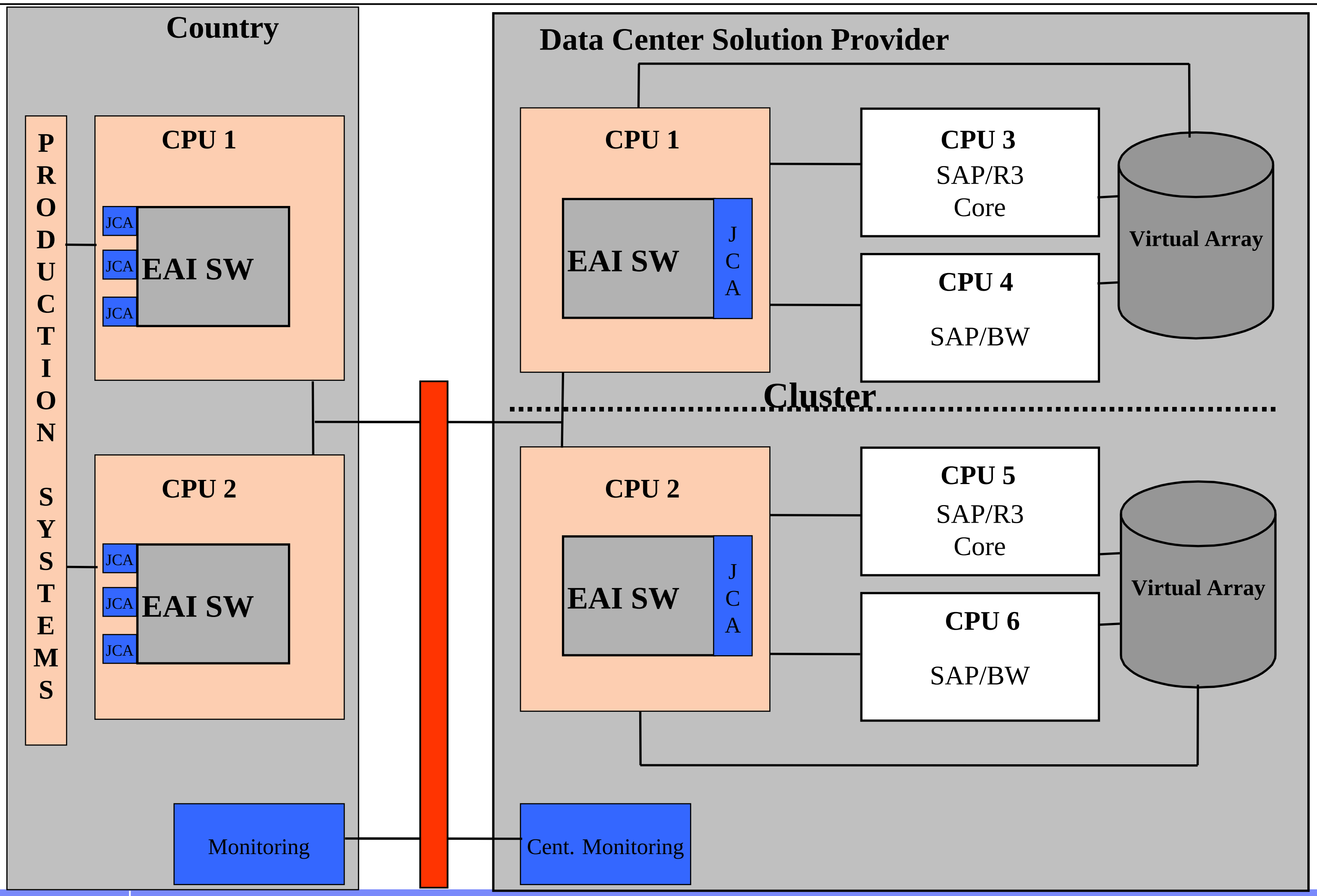
Examples of Application Landscapes

- Examples:
  - Operational systems in each company feed FI/CO system providing controlling functionality
  - Production planning on the corporate layer drives production in local plants
  - Consolidation of planning
  - Central procurement system
- Implications of new IT Strategies:
  - Implementing new business processes for reporting
  - Replacement of local FI/CO systems by a global solution
  - Centralization of IT support per country

Business and Application Landscape



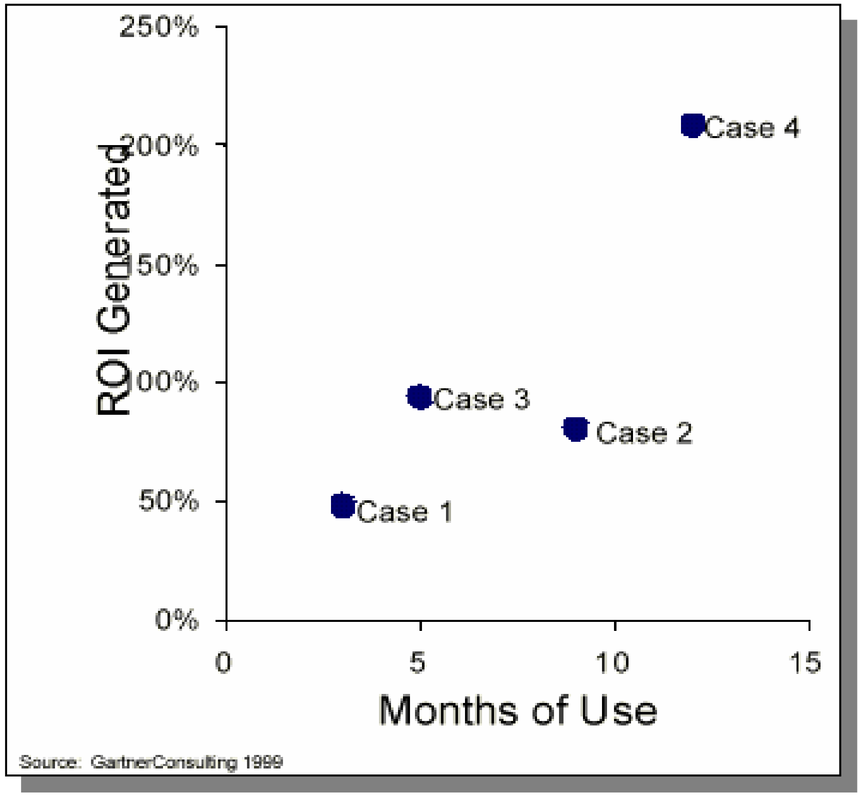
Example Pilot Implementation



Benefits

- Facilitates:
  - Integration of standard packages
  - Migrations
- General:
  - Reduction of development time approx. 20 - 40 %
  - Reduction of maintenance approx. 25 – 70 %
- Example:
  - 23 Applications: Reduction of devlopment time approx. 88%
  - ROI: 379%, Payback: 0.55 years

Gartner Report



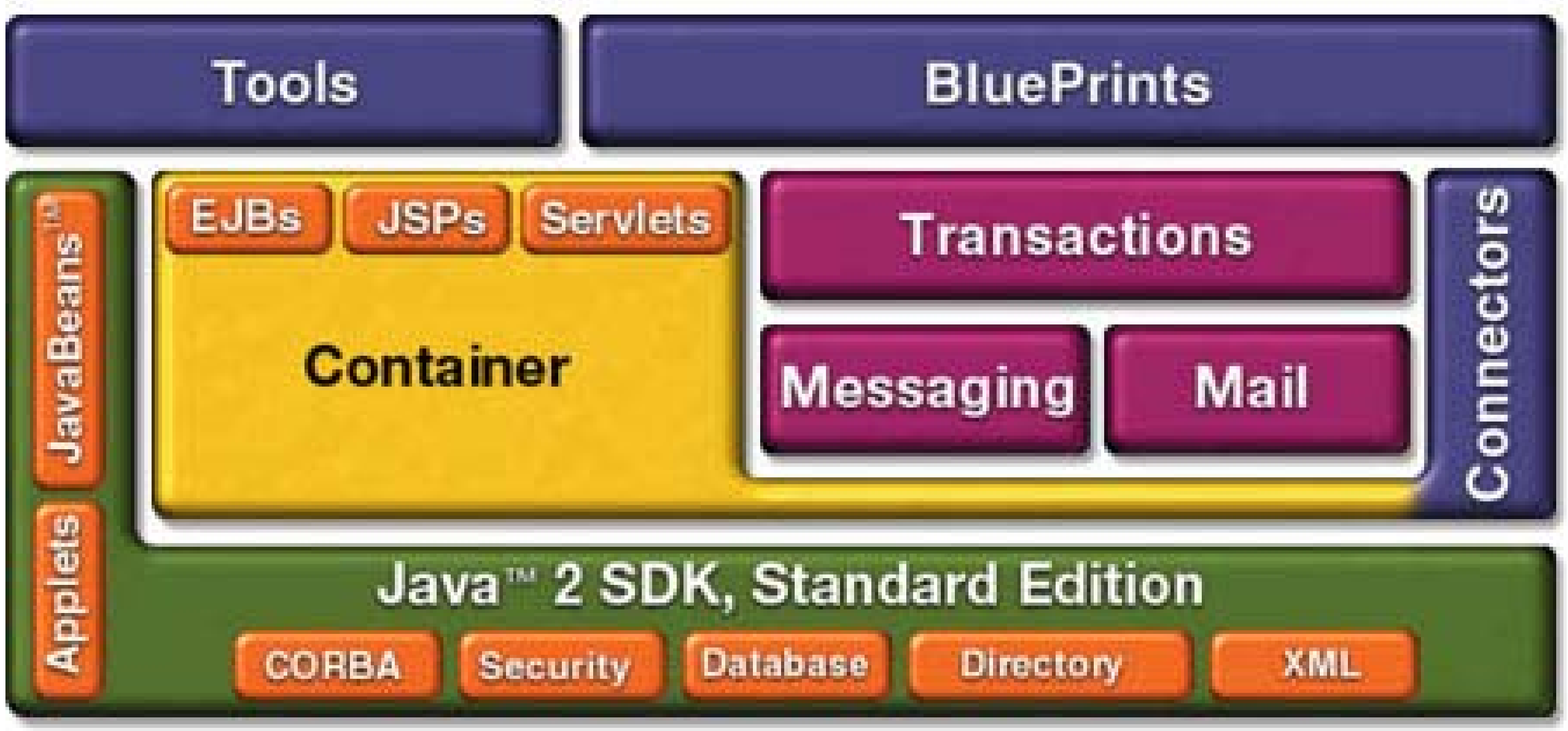
- *To summarize, the key drivers of ROI include the period of use and the complexity of organizations' environments.*
- *"We found quite consistently that these organizations broke even or nearly broke even in their first six to twelve months of using the* [EAI] *product.*
- *Their ROI varied by the amount of time using the integration broker and by the complexity of their operating environments.*
- *ROI ranged in value from 48% to 209%."*

III. EAI meets J2EE

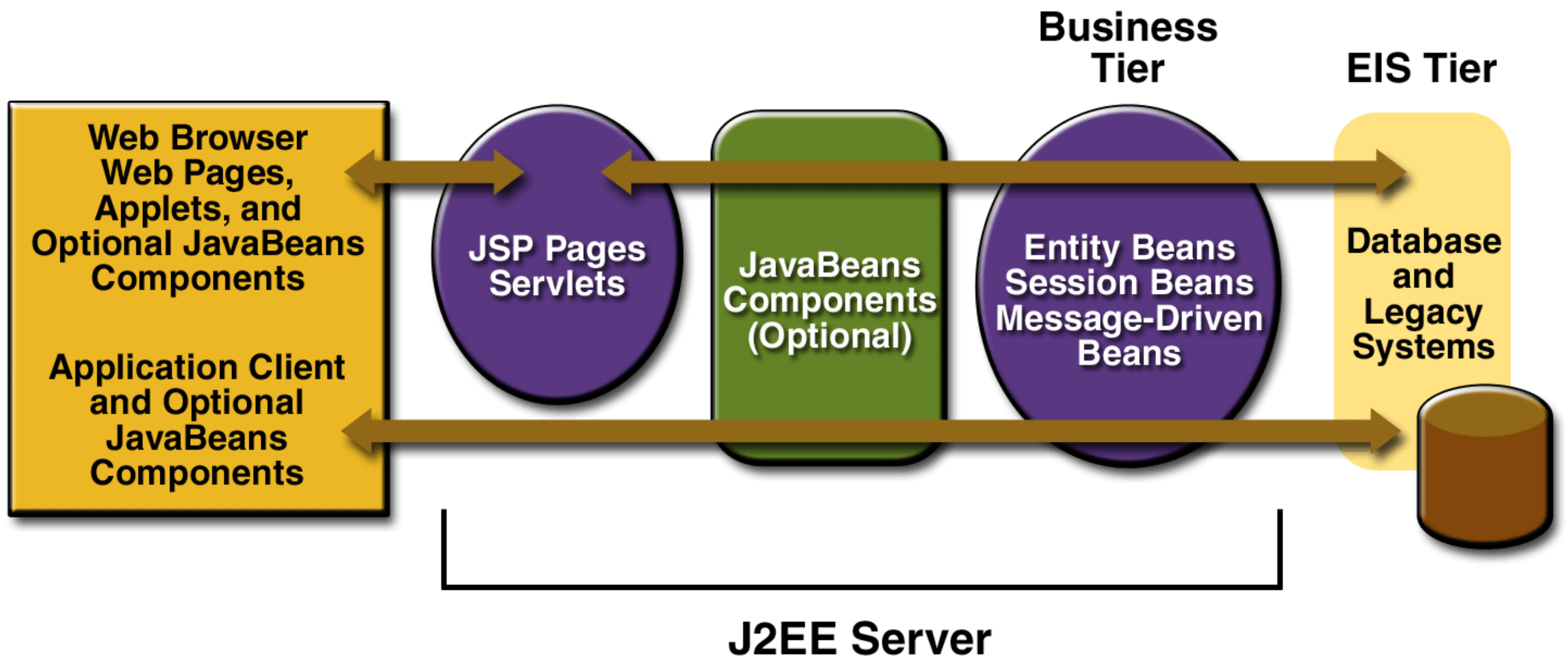
J2EE Framework

- Provides environment with Services:
  - Transaction
  - Messaging
  - …
- Provides architecture
  - JDBC (Java Database Connectivity)
  - JCA (Java Connector Architecture)

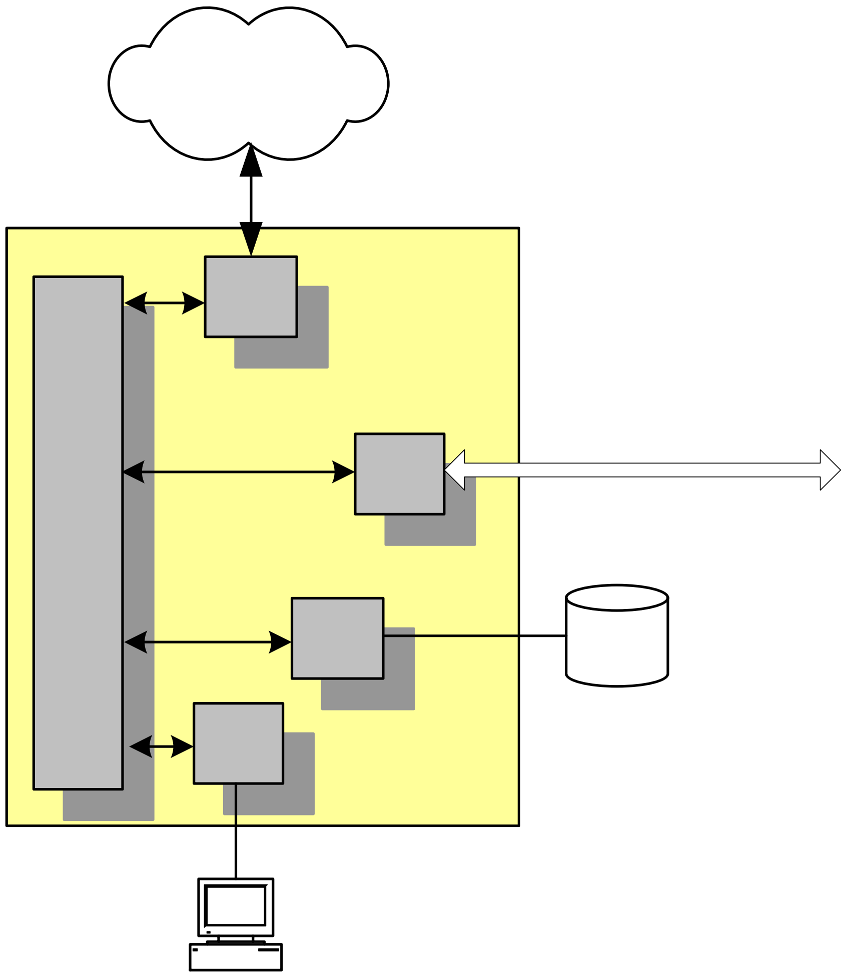
J2EE Services



J2EE Framework


Business Tier
EIS Tier
Web Browser Web Pages, Applets, and Optional JavaBeans Components
Application Client and Optional JavaBeans Components
JSP Pages Servlets
JavaBeans Components (Optional)
Entity Beans Session Beans Message-Driven Beans
Database and Legacy Systems
J2EE Server

Example: One Business System, Several Components



Result

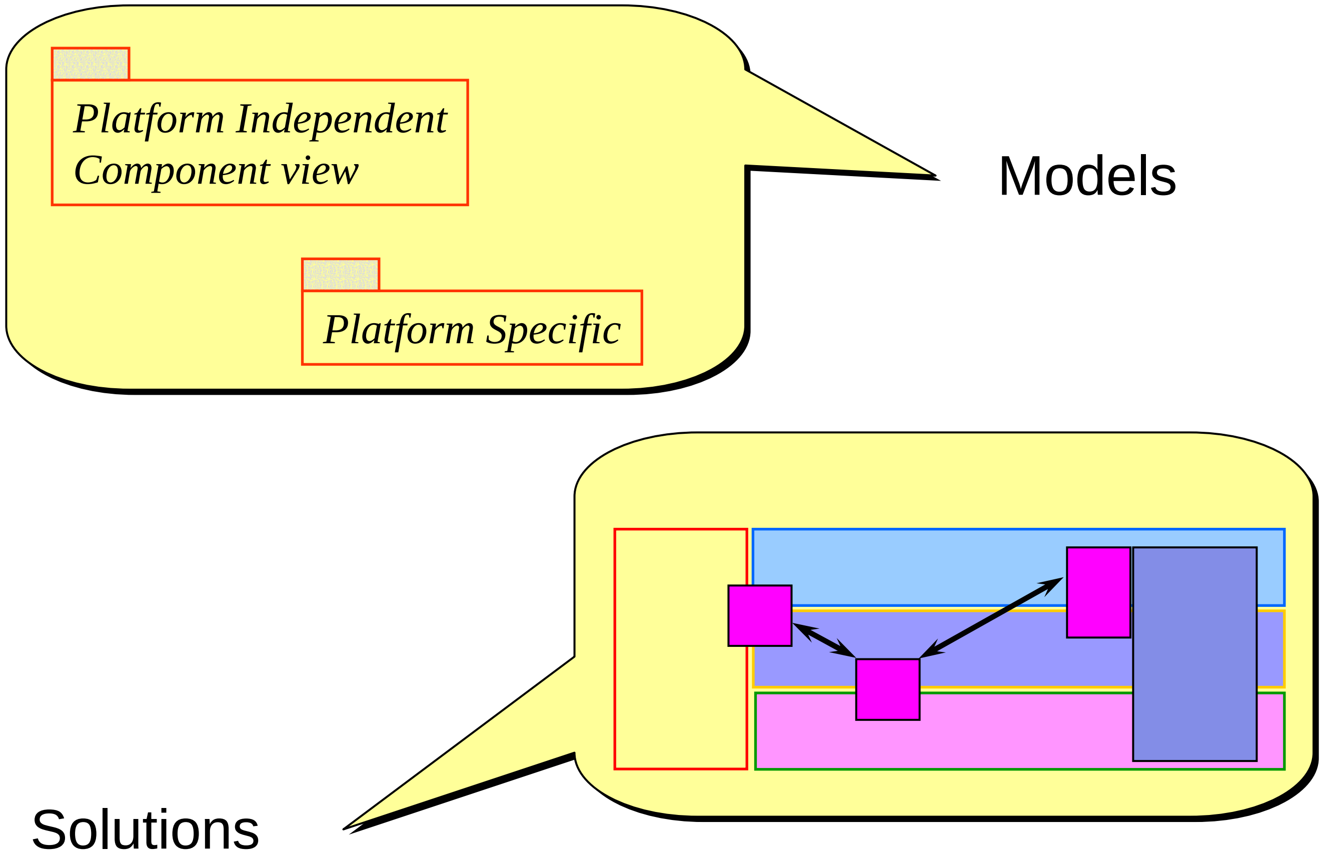
- Multiple complexities
  - Software development of components
  - Information flow
  - Data transformations
  - Multiple transactions
- Need to be managed
  - By an enterprise centric view
  - Appropriate levels of abstractions
  - Use of modeling
  - Apply MDA (Model Driven Architecture)

IV. Enterprise Centric View

An Enterprise Architecture has to provide Unified Views of

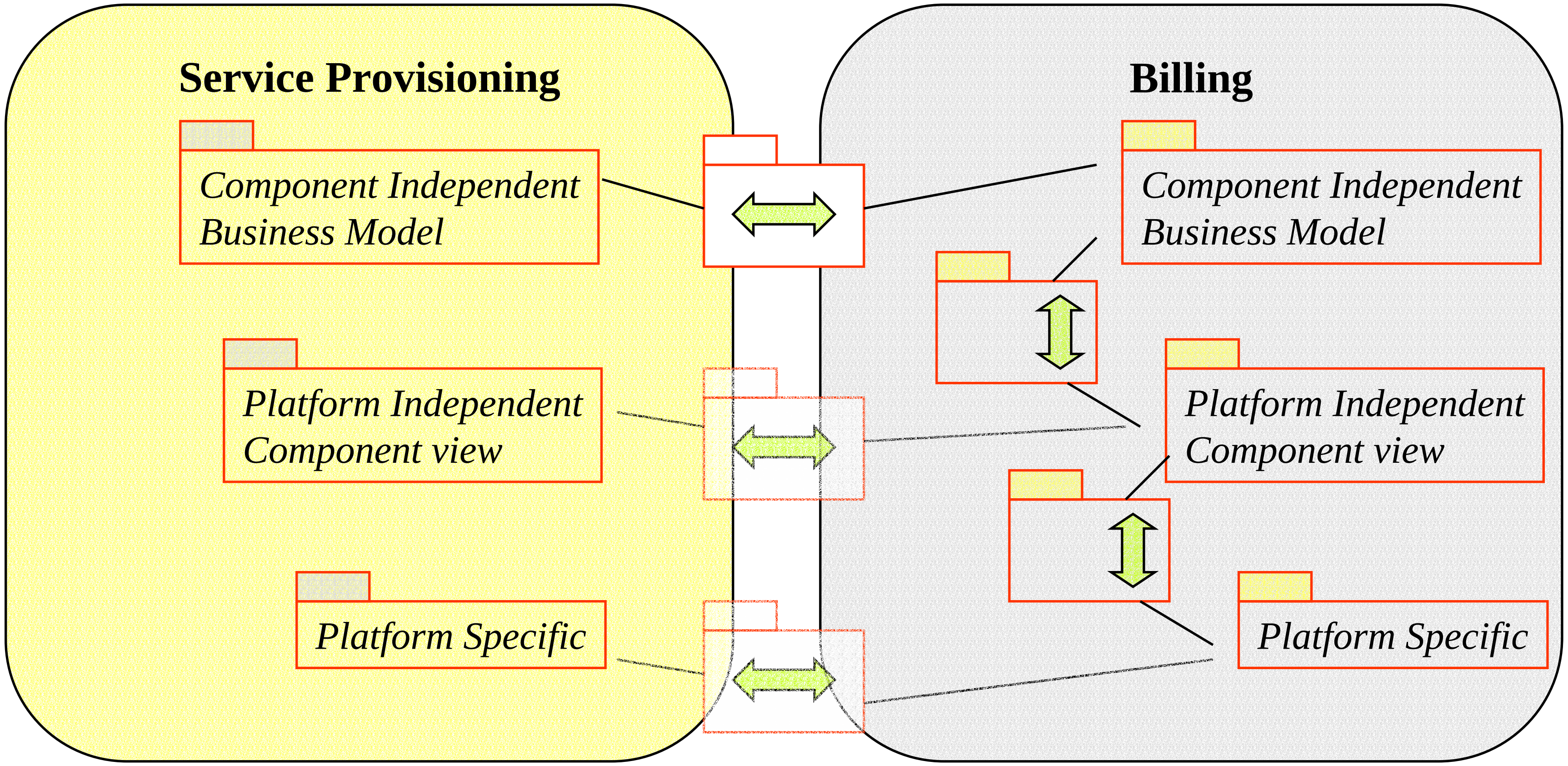
- Rules and Procedures
  - to capture business processes with the needed precision
- Semantics and ontologies
  - use the same terminology
- Service definitions
  - definitions of components and interfaces
  - access methods
- Addressing different abstraction layers
  - Hiding details of the implementation
  - Providing platform-independent view that is J2EE independent view (Going beyond)

Architectural Approach



Consistency by Modeling

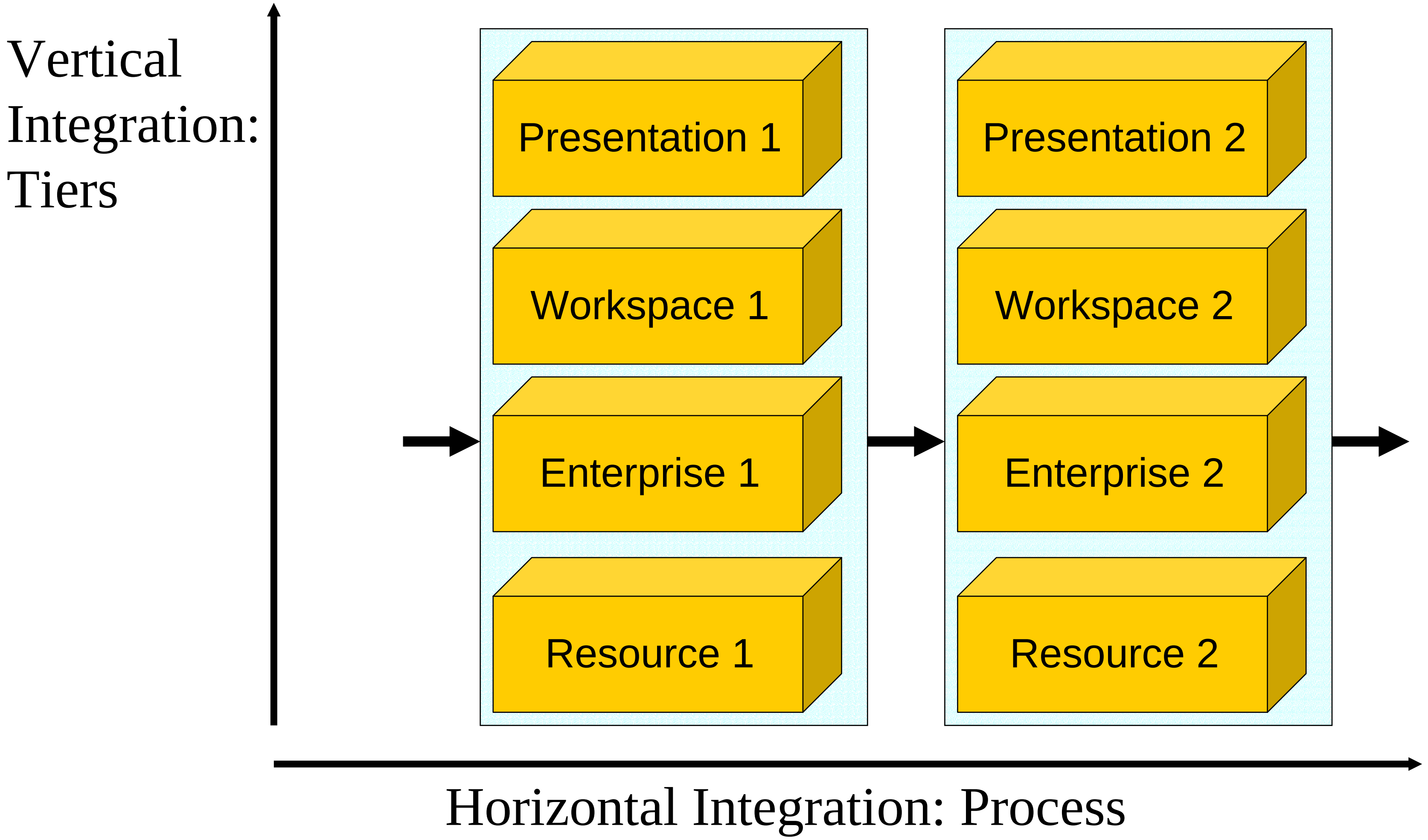
Mappings ⇕ and Transformations ⇔


Service Provisioning
Component Independent Business Model
Platform Independent Component view
Platform Specific
Billing
Component Independent Business Model
Platform Independent Component view
Platform Specific

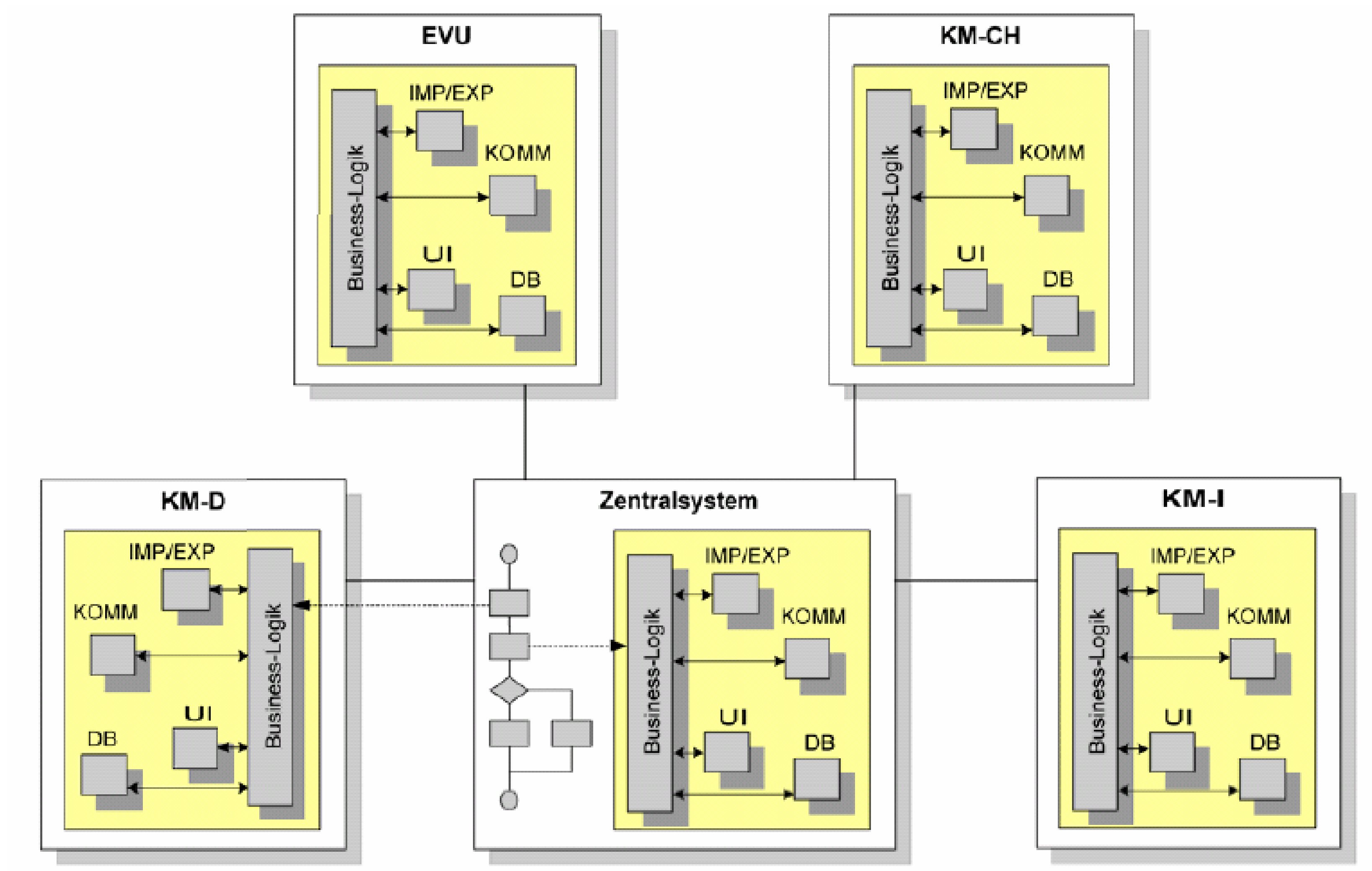
Main Aspects

- Business components arise at several tiers according to Herzum-Sims:
  - presentation,
  - workspace,
  - enterprise,
  - resource
- “Separation of Concerns”
- Every component can be seen on various abstraction levels
  - “platform-independent” – where the implementation technologies do no matter
  - “platform-dependent” – like WDSL, J2EE, …

Components: Process versus Tiers

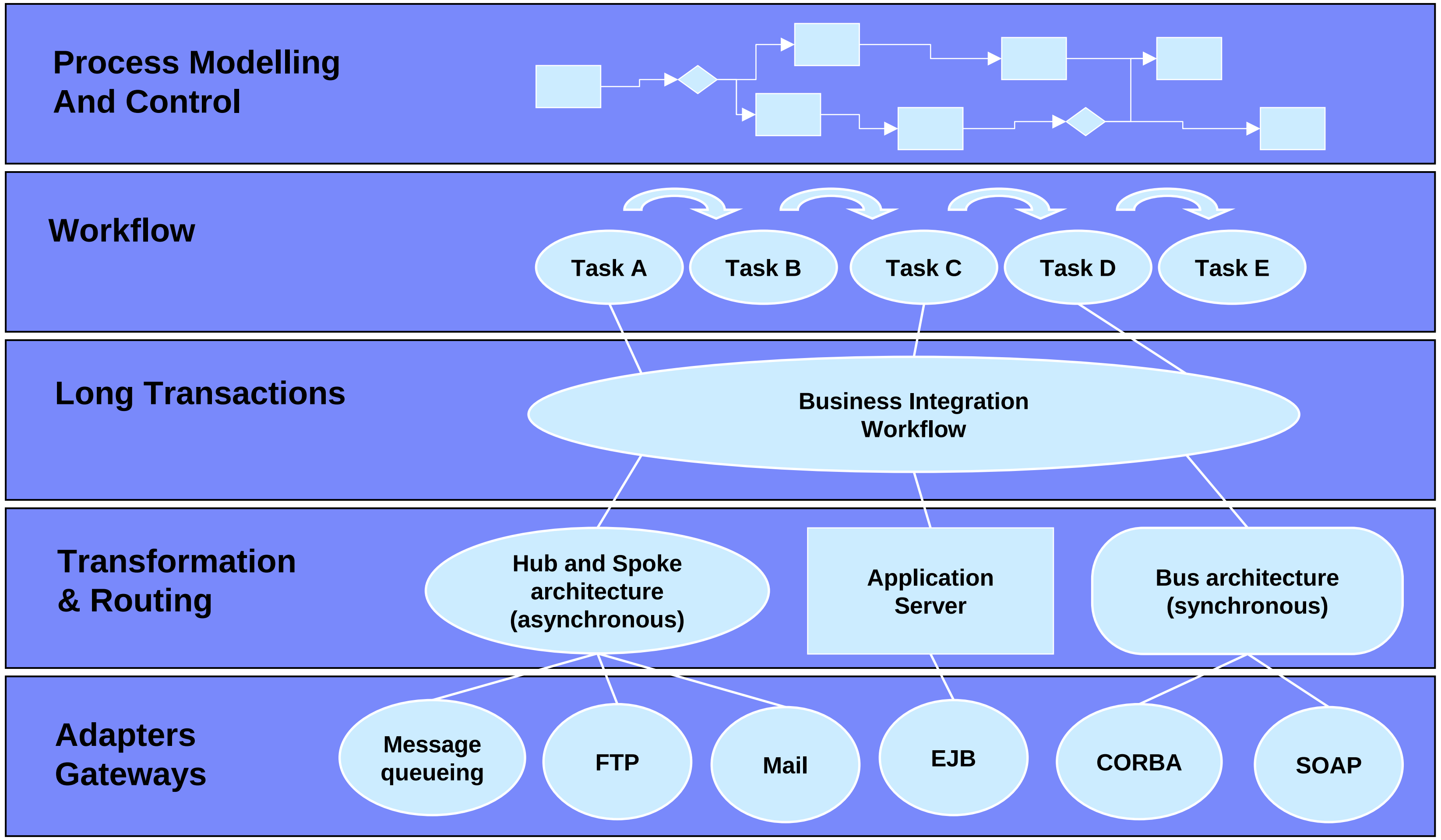


Components: Example Topology Multiple Systems

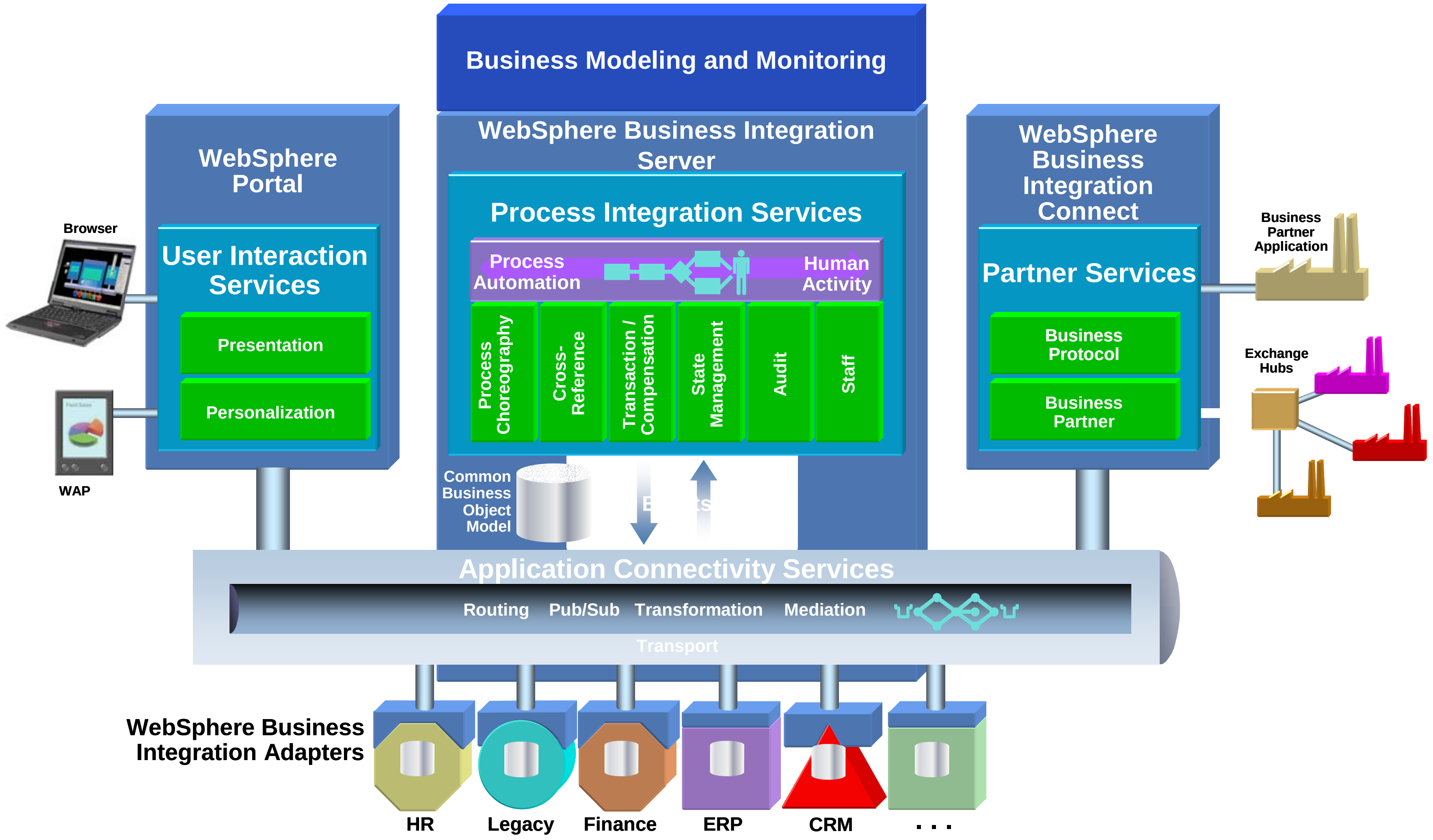


V. References

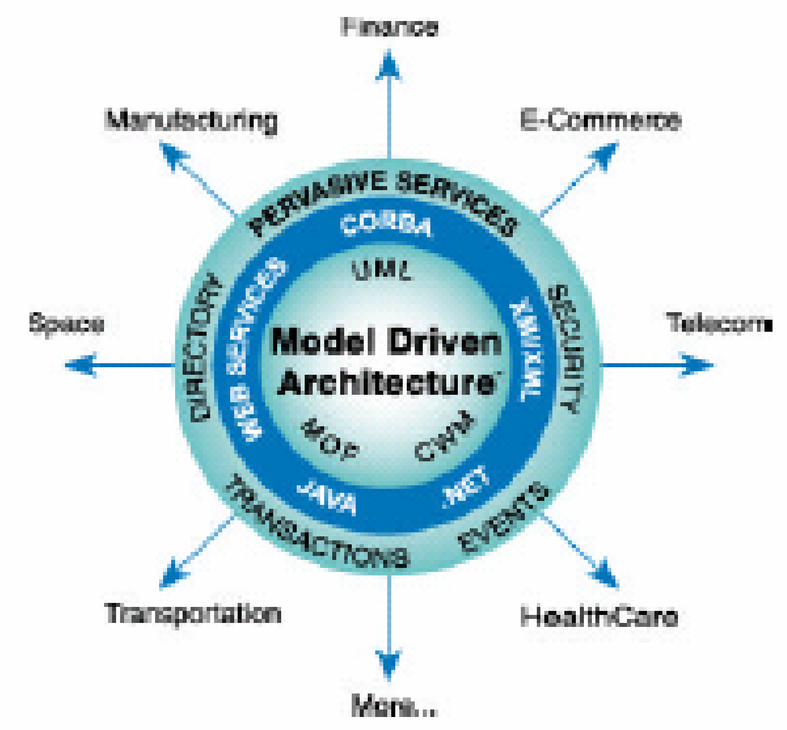
IBM Coverage of an Open and Modular EAI Architecture



IBM Websphere Enterprise Integration


Business Modeling and Monitoring
WebSphere Business Integration Server
Process Integration Services
Process Automation
Human Activity
Process Choreography
Cross-Reference
Transaction / Compensation
State Management
Audit
Staff
Common Business Object Model
Events
WebSphere Portal
User Interaction Services
Presentation
Personalization
Browser
WAP
WebSphere Business Integration Connect
Partner Services
Business Protocol
Business Partner
Business Partner Application
Exchange Hubs
Application Connectivity Services
Routing
Pub/Sub
Transformation
Mediation
Transport
WebSphere Business Integration Adapters
HR
Legacy
Finance
ERP
CRM
. . .

OMG's MDA (Model Driven Architecture)



- A set of standards defining the scope, contents, creation and usage of models
- An architecture-based method for integrating models into the development process
- Core Technologies
  - UML + OCL
  - MOF + XMI
  - CWM

Conclusion - Success Factors

- A holistic view is needed
- Technology is only a part of the picture
- Separation of abstraction layers (logical, technical)
- Stepwise approach
- IT matters (Response to Nicholas Carr)