Lesson 18 – Web Services and Service Oriented Architectures

Service Oriented Architectures Security

Module 4 - Architectures

Unit 1 – Architectural features

Ernesto Damiani

Università di Milano

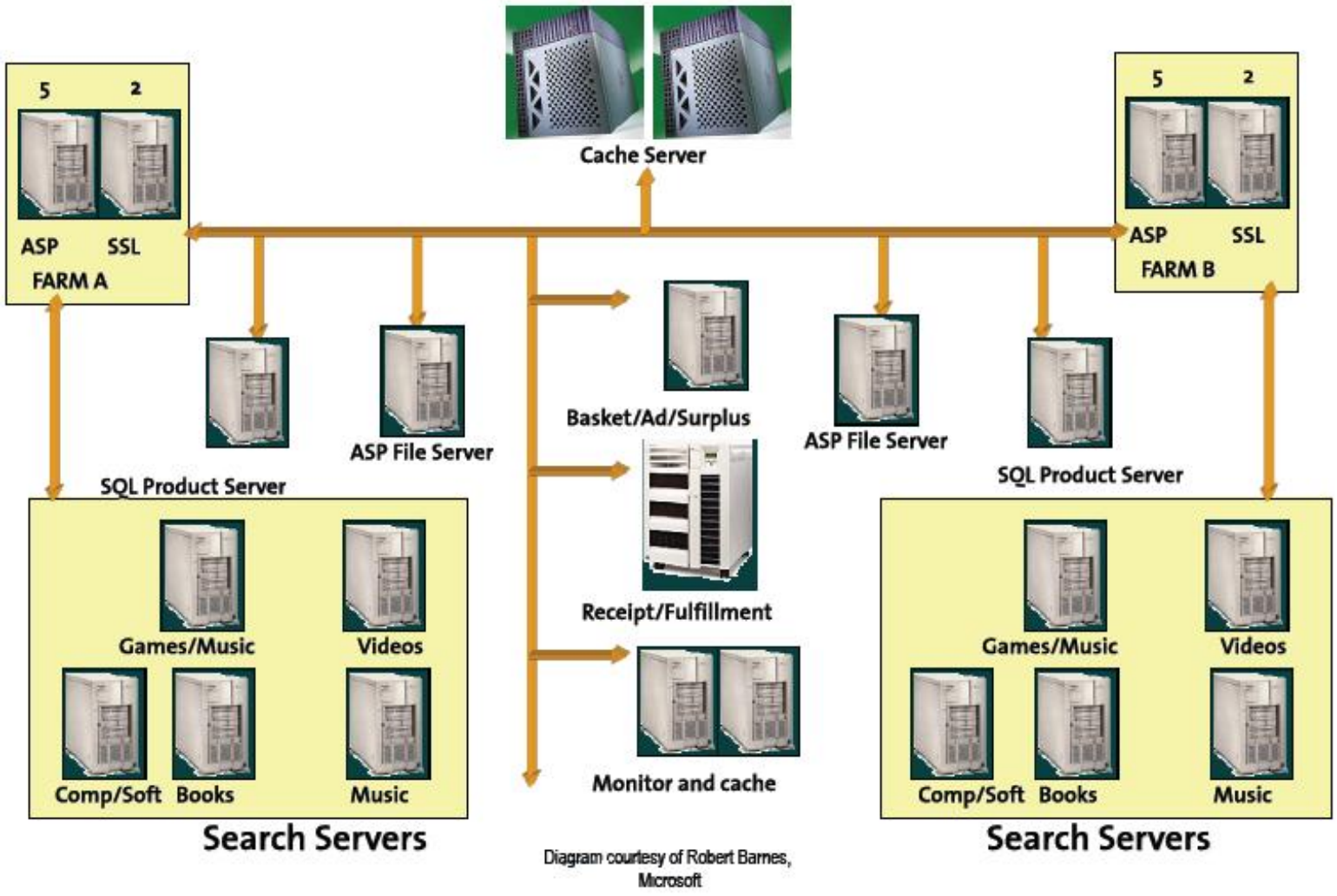
A bit of history (1)

- Enterprise computing has evolved along many directions in the last decade:
  - Data centers
  - Islands of information
  - Scale up vs. scale out
  - Networking and distribution
  - Internet and WWW
  - Middleware
  - RPC vs messaging
  - Programming languages

A bit of history (2)

- This is an evolutionary process with many changes and many aspects to it but with common themes:
  - Making enterprise IT more cost efficient and manageable
  - Making systems faster, more robust, more flexible
- We are in the middle of the evolution …

An e-commerce platform

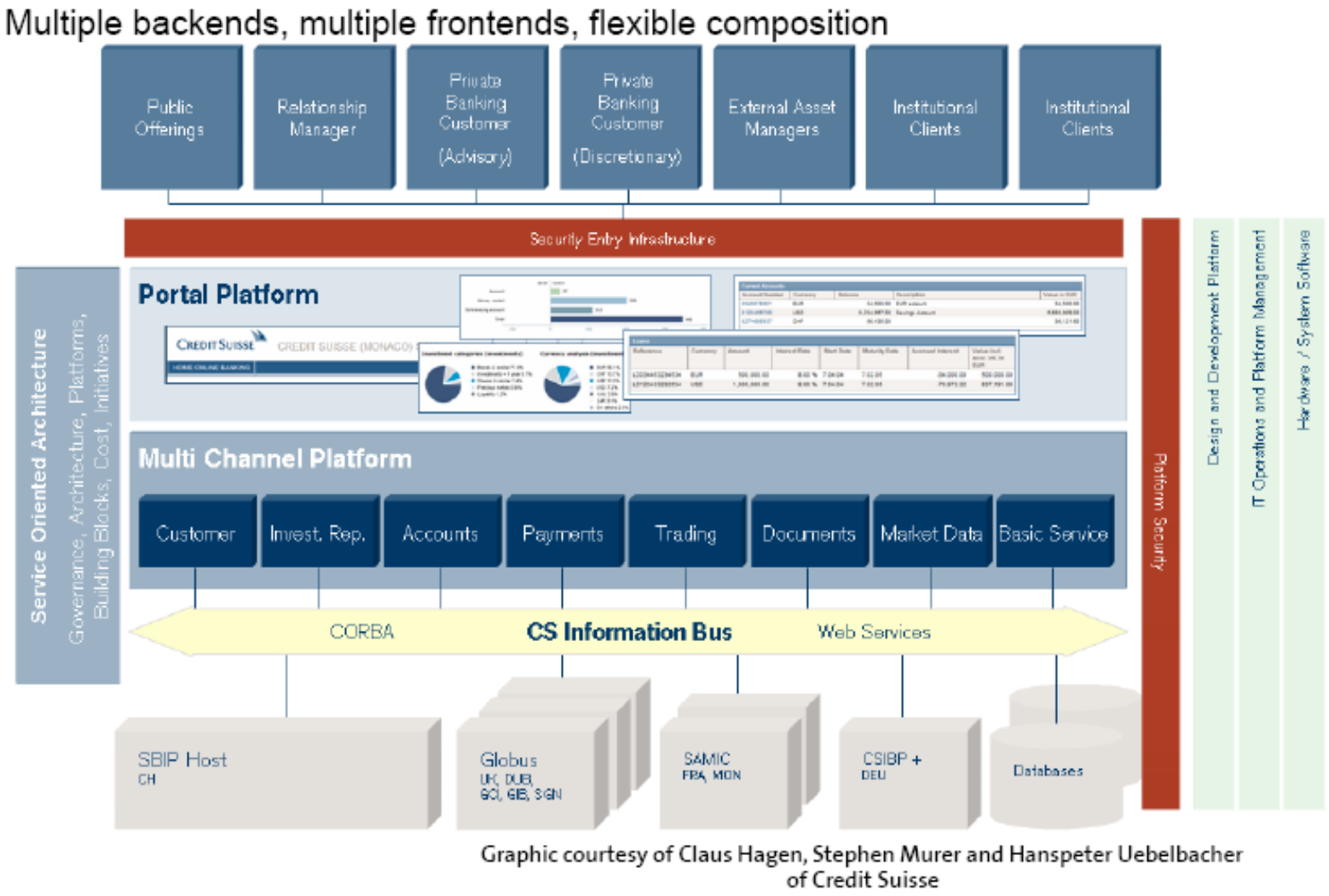


What is Enterprise Architecture?

Enterprise Architecture is the practice of applying a comprehensive and rigorous method for describing a current and/or future structure and behavior for an organization's processes, information systems, personnel and organizational sub-units, so that they align with the organization's core goals and strategic direction. Although often associated strictly with information technology, it relates more broadly to the practice of business optimization in that it addresses business architecture, performance management, organizational structure and process architecture as well.

- Wikipedia (http://en.wikipedia.org/wiki/Enterprise_architecture)

Enterprise Architecture at Credit Suisse



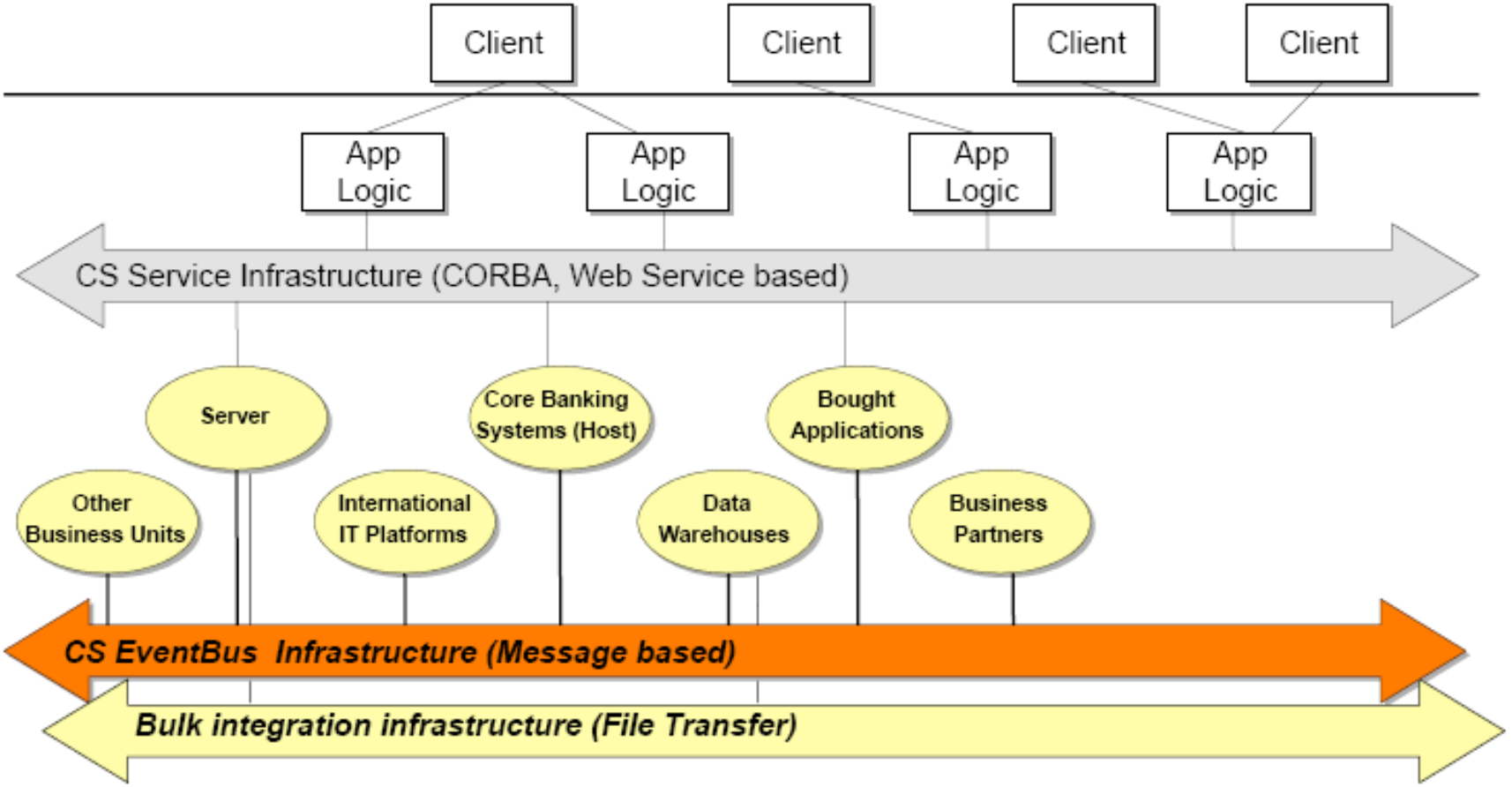
Graphic courtesy of Claus Hagen, Stephen Murer and Hanspeter Uebelbacher of Credit Suisse

Enterprise Application Integration

EAI (Enterprise Application Integration) is defined as the uses of software and computer systems architectural principles to integrate a set of enterprise computer applications

- Wikipedia (http://en.wikipedia.org/wiki/Enterprise_application_integration)

Integration Infrastructure - Credit Suisse

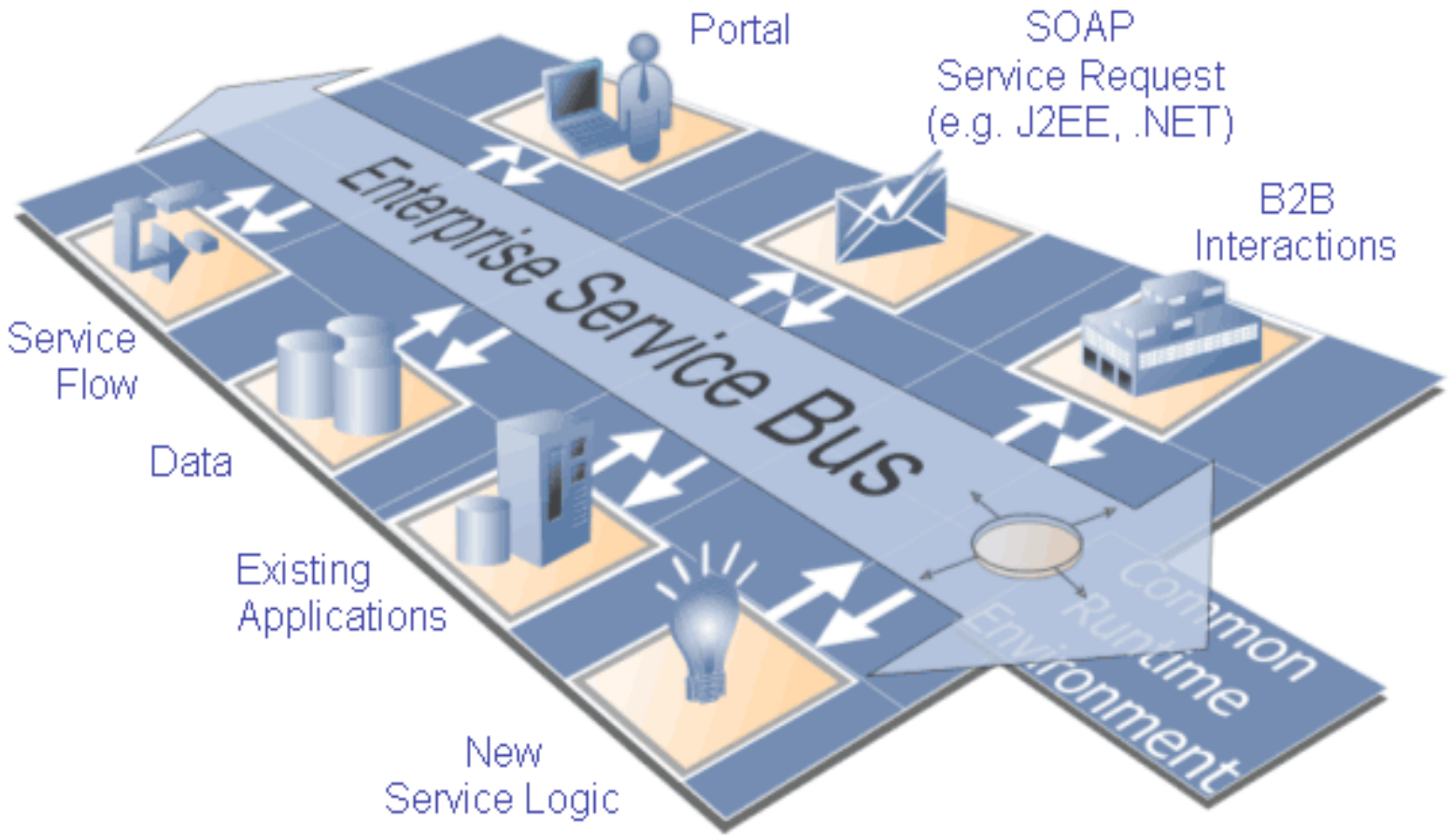


Graphic courtesy of Claus Hagen, Stephen Murer and Hanspeter Uebelbacher of Credit Suisse

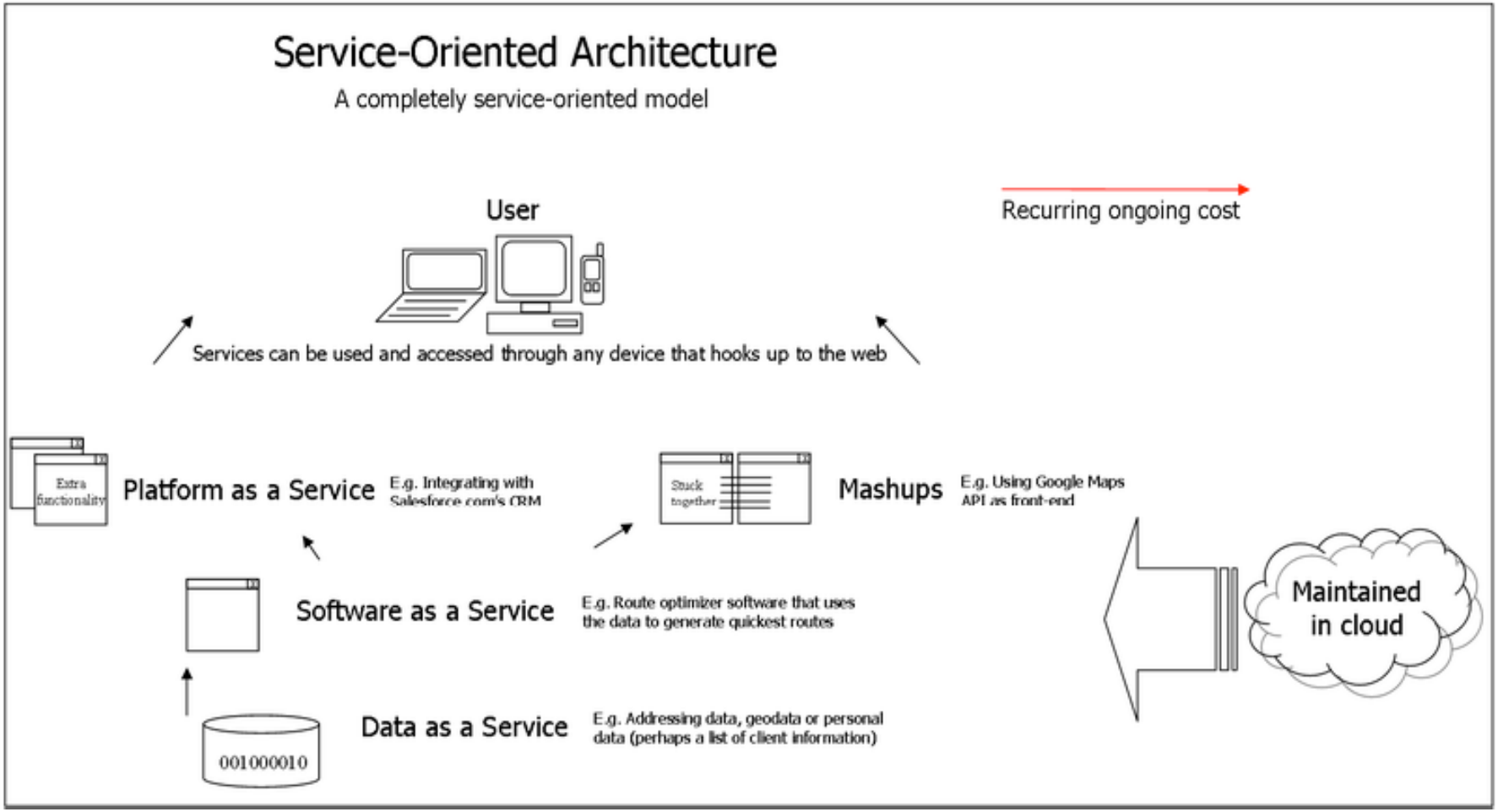
Many trends but one direction

- There are emerging patterns that can be identified from the marketing chaos:
  - loose coupling
  - messaging (instead of RPC)
  - software encapsulation at a larger scale (service)
  - need for governance
  - reliance on more complex infrastructures
- These elements are emphasized in different ways as technology evolves

An integration backbone



Cloud computing

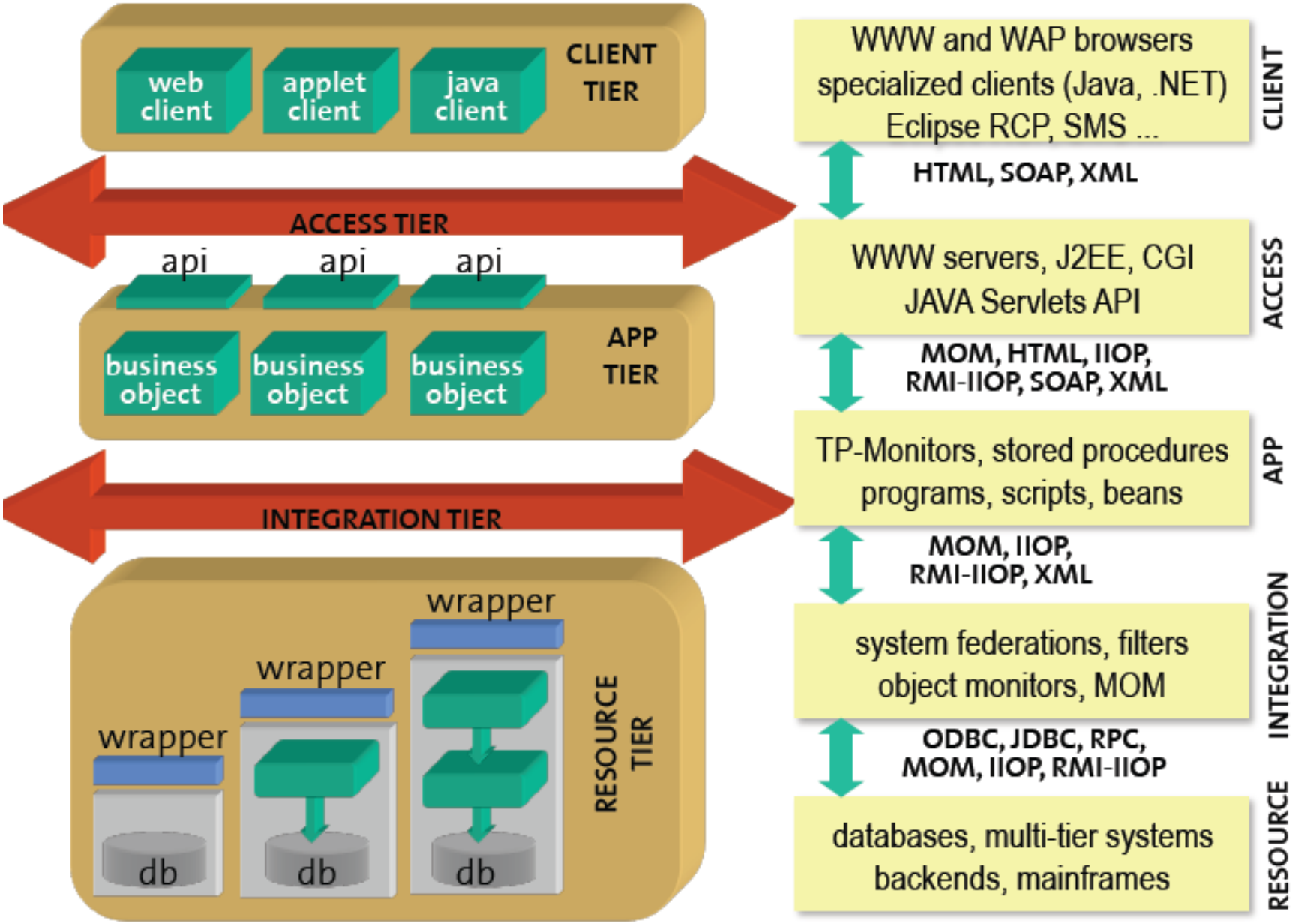


Imposing some order (1)

- For many years, enterprise computing was very confusing:
  - many products
  - many platforms
  - incompatibilities
  - many different approaches
  - field was not yet mature

Imposing some order (2)

- Service orientation is the first attempt at bringing some order
  - borrowing concepts from software engineering
  - making services the unit of discourse
  - understanding architectures
  - understanding the key concepts


web client
applet client
java client
CLIENT TIER
ACCESS TIER
api
api
api
business object
business object
business object
APP TIER
INTEGRATION TIER
wrapper
wrapper
wrapper
db
db
db
RESOURCE TIER
WWW and WAP browsers
specialized clients (Java, .NET)
Eclipse RCP, SMS ...
CLIENT
HTML, SOAP, XML
WWW servers, J2EE, CGI
JAVA Servlets API
ACCESS
MOM, HTML, IIOP,
RMI-IIOP, SOAP, XML
TP-Monitors, stored procedures
programs, scripts, beans
APP
MOM, IIOP,
RMI-IIOP, XML
system federations, filters
object monitors, MOM
INTEGRATION
ODBC, JDBC, RPC,
MOM, IIOP, RMI-IIOP
databases, multi-tier systems
backends, mainframes
RESOURCE

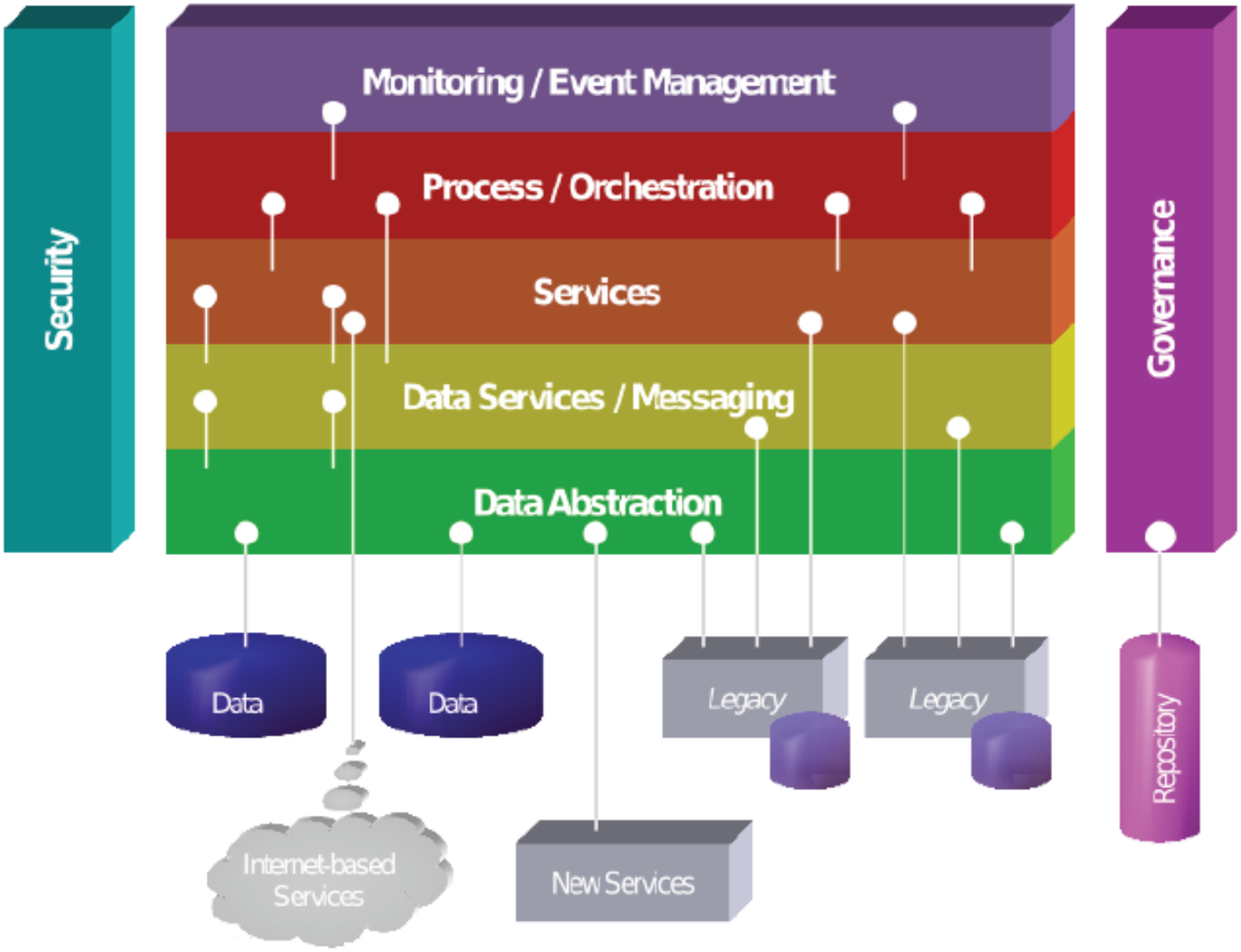
What is SOA?

- SOA = Services Oriented Architecture
  - Services = another name for large scale components wrapped behind a standard interface (Web services although not only)
  - Architecture = SOA is intended as a way to build applications and follows on previous ideas such as software bus, IT backbone, or enterprise bus
- The part that it is not in the name
  - Loosely-coupled = the services are independent of each other, heterogeneous, distributed
  - Message based = interaction is through message exchanges rather than through direct calls (unlike Web services, CORBA, RPC, etc.)

What SOA is about?

- Conventional software engineering cannot address:
  - Non-functional properties of complex distributed systems: performance, reliability, security, …
  - Large scale composition
  - Document and message based interaction
  - Distributed systems architecture
  - Run time properties and contracts
  - Protocols, data formats
  - Reusability
- SOA and web services bring to the fore the dependencies of existing programming languages on (single CPU) hardware concepts

SOA metamodel



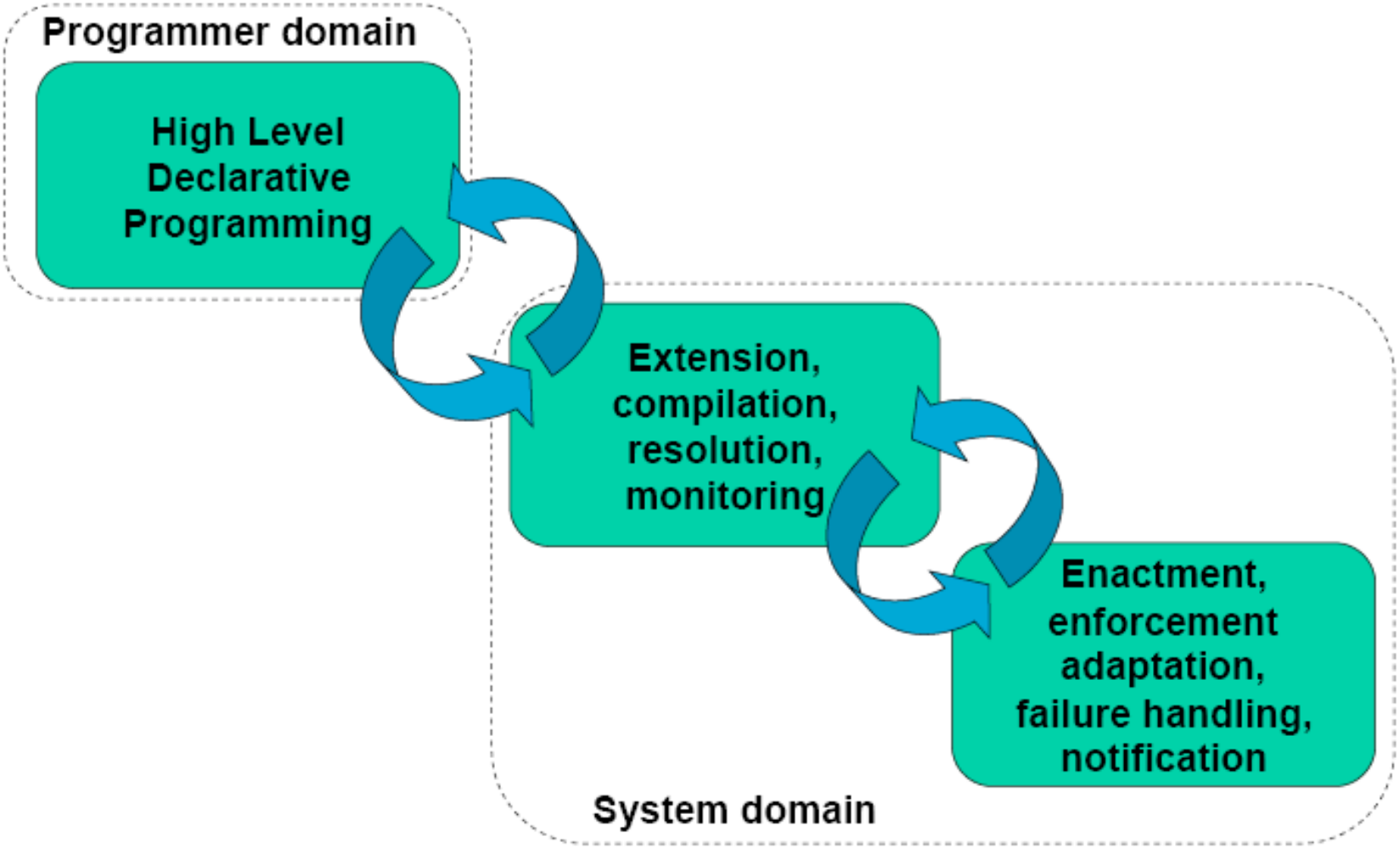
The novelty behind SOA (1)

- The concept of SOA is not new:
  - message oriented middleware
  - message brokers
  - event based architectures
- The current context is different
  - emergence of standard interfaces (Web services)
  - emphasis on simplifying development (automatic)
  - use of complex underlying infrastructure (containers, middleware stacks, etc.)

The novelty behind SOA (2)

- Interest in SOA arises from a number of reasons:
  - basic technology in place
  - more clear understanding of distributed applications
  - the key problem is integration not programming

Automatic Software Generation



System integration

- SOA is not an abstract idea, it is a response to
  - Very well defined needs in enterprise computing
  - Changes in technology (cluster, networking, distribution)
  - The Internet
  - Business-to-Business integration
  - Enterprise Computing coming of age
  - Large scale distributed information systems
- To understand SOA, the key is to understand the underlying problems, not just the technology involved in solving them

Automatic Plumbing Generation (1)

- The promise of SOA is to facilitate integration by
  - letting the system automatically decide on integration decisions
    - protocols to use
    - intermediate processing needed
    - routing and distribution
    - data transformations
  - enforcing standards for defining and operating with services
  - enforcing the use of service interfaces

Automatic Plumbing Generation (2)

- Related to this, there are many additional opportunities:
  - languages for orchestration and composition
  - reasoning about compositions
  - integration by non-experts (IBM’s situational integration)

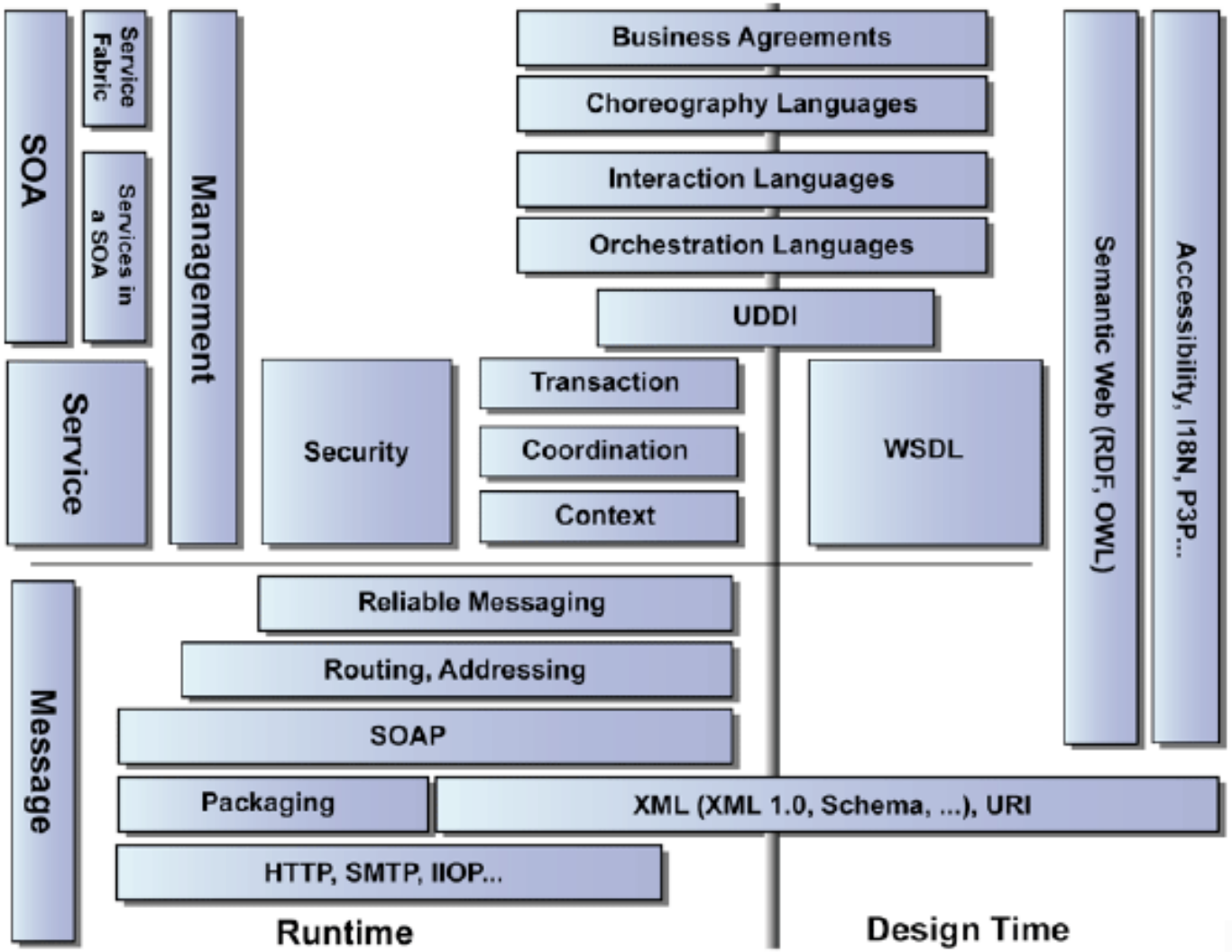
Example of easier integration

- WS Invocation Framework
  - Use WSDL to describe a service
  - Use WSIF to let the system decide what to do when the service is invoked:
    - if the call is to a local EJB then do nothing
    - if the call is to a remote EJB then use RMI
    - if the call is to a queue then use JMS
    - if the call is to a remote Web service then use SOAP and XML
  - There is a single interface description, the system decides on the binding
  - This type of functionality is at the core of the notion of Service Oriented Architecture

WS standards

Business Domain Specific extensions	Various	Business Domain
Distributed Management	WSDM, WS-Manageability	Management
Provisioning	WS-Provisioning	
Security	WS-Security	Security
Security Policy	WS-SecurityPolicy	
Secure Conversation	WS-SecureConversation	
Trusted Message	WS-Trust	
Federated Identity	WS-Federation	
Portal and Presentation	WSRP	Portal and Presentation
Asynchronous Services	ASAP	Transactions and Business Process
Transaction	WS-Transactions, WS-Coordination, WS-CAF	
Orchestration	BPEL4WS, WS-CDL	
Events and Notification	WS-Eventing, WS-Notification	Messaging
Multiple message Sessions	WS-Enumeration, WS-Transfer	
Routing/Addressing	WS-Addressing, WS-MessageDelivery	
Reliable Messaging	WS-ReliableMessaging, WS-Reliability	
Message Packaging	SOAP, MTOM	
Publication and Discovery	UDDI, WSIL	Metadata
Policy	WS-Policy, WS-PolicyAssertions	
Base Service and Message Description	WSDL	
Metadata Retrieval	WS-MetadataExchange	

Separation of concerns



FINE