Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)

Roberto Bruni

http://www.di.unipi.it/~bruni

06 - Evolution
Overview of the evolution of (Information Systems inside) Enterprise Systems Architectures

Ch.2 of Business Process Management: Concepts, Languages, Architectures
Guiding principles

Modularity and information hiding
(encapsulation, interfaces, reuse, maintainability, response to change)
Software Architecture

Definition: A software architecture defines a structure that organizes the software elements and the resources of a software system.
Gartner’s hype cycle

A **hype cycle** is a (branded) graphic representation of the maturity, adoption and social application of specific technologies.
Early systems (architectures)

- Monolithic applications developed from scratch
- Porting required redevelopment
- Data dependency and consistency issues

1970
- Application
- Programming interfaces
- OS

1980
- Application
- Physical data independence
- DBMS
- OS

1990
- GUI
- Advanced user interfaces
- Application
- Data management as a primary concern
- Database

Application code and (textual) user interfaces still entangled
Data dependency and consistency issues
Human interaction made easier

Enterprise Applications

OS + DBMS + GUI + Networking capabilities = more and more elaborate information systems could be engineered

Typically hosting enterprise applications (customers, personnel, products, resources)

Next steps: from individual to multiple information systems (needs integration)
Individual enterprise application

Lack of Integration!

Data redundancy!

Data dependencies!
Changes

Changes were hard to implement!

Hard to track data dependency and replication

Any modification of an application was a complex and error-prone activity, with domino effect (e.g. change of customer address format)
ERP

Enterprise Resource Planning (ERP) systems were developed to deal with the increasing complexity of changes

Basic idea
integrated database that spans most applications, separated modules provide desired functionalities, accessed by client applications
Enterprise resource planning systems

- Integrated and consistent (centralized) database
- Two-tier client-service
- Remote data access

ERP Server Application
- DBMS
- OS

Client 1

Client 2

... Client n

human resources

financials

manufacturing

ERP
CRM and SCM

New types of sw entered the market around 2000

**Customer Relationship Management (CRM) systems**
**Supply Chain Management (SCM) systems**

**Goal**

to support the planning, operation, and control of supply chains, including inventory management, warehouse management, management of suppliers and distributors, and demand planning

**Problem**: different vendors, separately developed
Siloed enterprise applications

Data Integration would provide valuable information

Customer Relationship Management System

Supply Chain Management System

Lack of Integration!

Data redundancy!

Data dependencies!

(on a larger scale and complexity than before)

Connected on local network, but not logically integrated
A sample scenario

Customer calls

Call centre personnel can only access the information stored in one system

Call centre personnel is not aware of the full status of the customer

Customer (doesn’t care about siloed structure) does not feel well served, becomes upset, expects a better service
Heterogeneity

Heterogeneity of data and their attributes
(syntax and semantics difficulties)
calls for Data Integration

Examples

corresponding data fields with different names
(e.g., CustAddr vs CAstreeet),

fields with the same name but different meaning
(e.g. Price: with or without taxes? unitary?)
Integration

Manual integration is possible, but:

- it consumes considerable resources
- it is error-prone
- cannot be foreseen all applications in advance
  (reimplementing functionalities in an integrated way
  would just postpone the problem)

Solution

Enterprise Application Integration systems
as a new middleware
Enterprise Application Integration

Definition: **Enterprise Application Integration** (EAI) is defined as the use of software and computer systems architectural principles to integrate a set of enterprise computer applications.
Point-to-point integration (of silos)

N x N hard-wiring problem!

Too many interfaces to develop!

Does not respond well to changes!
(Reprogramming an interface requires considerable resources, typically)

\[ \sum_{i=1}^{N-1} i = \frac{N(N - 1)}{2} \]
Message Oriented Middleware

**Message-Oriented Middleware** offers some execution guarantees, such as message delivery (e.g. persistent message queues are used)

Still, the main problem remains: changes in the application landscape require changes in the communication structure

The Client exploits an **Integration Application** to operate on all systems
Message-oriented middleware

Cooperation realized in the integration application

Messages must be encoded and decoded

Point-to-point connection problem does not diminish that much
Response to Change

Message-oriented middleware reduces in part integration efforts and gives important run-time guarantees.

Still cooperation is hardwired in a particular application (the Integration Application).

No explicit process model that can be documented, communicated, and changed when necessary.

In the end, response to change is not improved.
The **Hub-and-Spoke** paradigm is based on a central hub and a number of spokes attached to it.

The Application Integration middleware represents the hub, and the applications to be integrated represent the spokes.

Interactions between any two applications must pass through the hub.
Hub-and-spoke integration

Configuration and management of adapters and message brokers can become cumbersome.

From N x N to N integrators
Message brokers
Publish/subscribe mechanism

ERP System
Centralized Enterprise Application Integration Middleware (Hub)
CRM System
SCM System
Inventory Management
Data Warehouse
Human Resources Application

EAI implementation pitfalls

70% of all EAI projects fail (2003). Most of these failures are not due to technical difficulties, but due to management issues:

- Constant change
- Shortage of EAI experts
- Competing standards
- Loss of detail: Information unimportant at an earlier stage may become crucial later
- Conflicting and emerging requirements
- Data protectionism
From (data-models and) data-integration

To (process-models and) process-integration
Value Chains and Process Orientation

Two major factors fuelled business process management

Value chains
as a means to functionally break down the activities a company performs

Process orientation
as the way to organize the activities of enterprises
Definition: a single-application workflow consists of activities and their causal and temporal ordering that are realized by one common application system.
Multiple-application workflow system

**Definition:** a multiple-application workflow contains activities that are realized by multiple application systems, providing an integration of these systems.
System workflow

Definition: a system workflow consists of activities that are implemented by software systems without any user involvement.
Workflows fit well with hub-and-spokes EAI
Limitations in workflow management

Technical integration problems:

Scarcely documented applications

Different levels of granularity

Tight coupling of applications (direct invocation)
Enterprise service computing

Main idea:

Business functionalities exposed as services

Services are equipped with usage information

Customers can find services and use them
Services

Definition: **Services** are loosely-coupled computing tasks that can be dynamically discovered and invoked over the network.

Each service comes with a **service description** that can be published in **service registries** by the **service provider**.

Service registries can be queried by **service requestors**.

Service descriptions provide a level of detail that facilitates service requestors to **bind** and **invoke** them.
Service-oriented architectures

Definition: **Service-oriented architectures** (SOA) are software architectures that provide an environment for describing and finding software services, and for binding to services.
Service-oriented architectures
Service enabled application system

standard interface

ERP Enterprise Services

service specification must be decoupled from implementation and legacy system

ERP System

DBMS

OS

ERP Database

Composite service based application

Intra-company well-expressed as business processes

Local registry
Manual search (absence of dynamic matchmaking)
Advantages of SOA

Reuse of functionality at coarse level of granularity

New applications can be built with less effort

Existing applications can be efficiently adapted to changing requirements

Reduced maintenance and development costs
Products as services

Corporations are increasingly perceived by the set of services they provide

These services exposed to the market can be realized by enterprise services (provided by the back-end application system)

Also services provided by third parties can be integrated so that better end used services can be provided to the customer
Business-to-business value system
Business-to-business processes