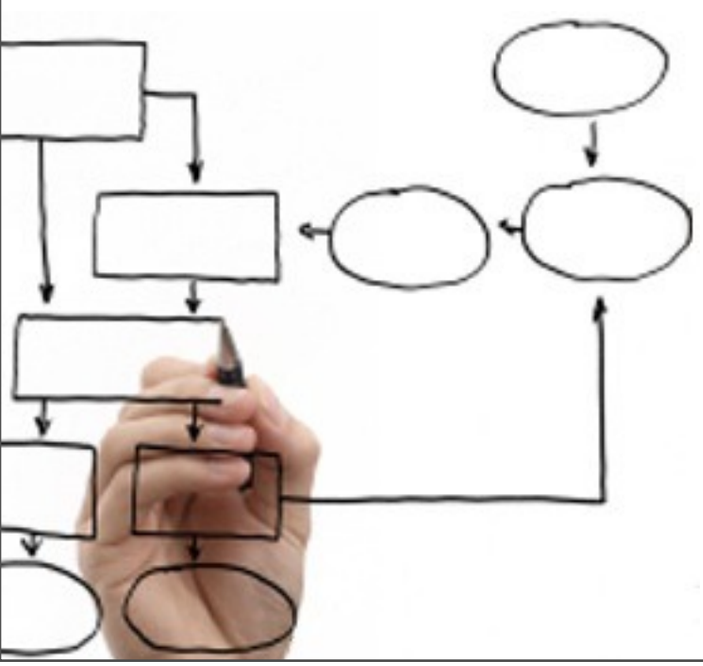
Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)



Roberto Bruni

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

05 - Evolution

Object

Overview of the evolution of
(Information Systems inside)
Enterprise Systems Architectures

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

Guiding principles

Separation of concerns
(to separate a system into distinct features that overlap in functionality as little as possible)

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

SoC: an example

HyperText Markup Language (HTML):
organization of webpage content

Cascading Style Sheets (CSS):
definition of content presentation style

JavaScript (JS):
user interactions

SoC: an example

Model–view–controller (MVC) sw architecture

Controller: send commands to the model to update the model's state or to its associated view to change the view's presentation of the model

Model: notifies its associated views and controllers when there has been a change in its state (the views update their output, the controllers change the available set of commands).

View: requests information from the model to generate an output representation to the user

Edsger W. Dijkstra
1974

11 May 1930
6 August 2002
http://www.cs.utexas.edu/users/EWD/

Let me try to explain to you, what to my taste is characteristic for all intelligent thinking.

It is, that one is willing to study in depth an aspect of one's subject matter ***in isolation for the sake of its own consistency****, all the time knowing that one is occupying oneself only with one of the aspects.*

Edsger W. Dijkstra
1974

...
*We know that a program must be **correct** and we can study it from that viewpoint only;*
*we also know that it should be **efficient** and we can study its efficiency on another day, so to speak.*
*In another mood we may ask ourselves whether, and if so: why, the program is **desirable**.*

But nothing is gained —on the contrary!— by tackling these various aspects simultaneously.

Edsger W. Dijkstra
1974

...
It is what I sometimes have called ***"the separation of concerns"****, which, even if not perfectly possible, is yet the only available technique for effective ordering of one's thoughts, that I know of.*

This is what I mean by "focussing one's attention upon some aspect": it does not mean ignoring the other aspects, it is just doing justice to the fact that ***from this aspect's point of view,***
the other is irrelevant.

Edsger W. Dijkstra
1974

Business data processing systems are sufficiently complicated to require such a separation of concerns

and the suggestion that in that part of the computing world "scientific thought is a non-applicable luxury" puts the cart before the horse: the mess they are in has been caused by ***too much unscientific thought****....*

Software Architecture

Definition: A **software architecture** defines a structure that organizes the software elements and the resources of a software system (outside view).

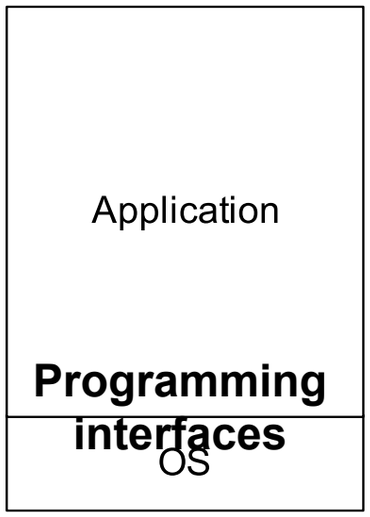
Software elements and resources are represented by subsystems, with specific responsibilities and relationships (inside view).

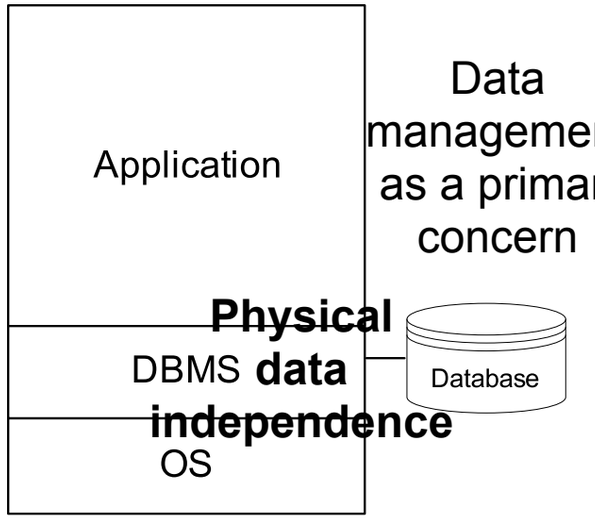
Early systems (architectures)

Applications developed from scratch

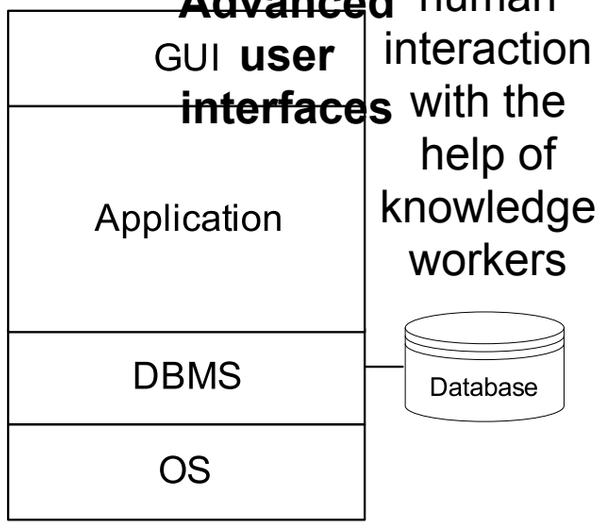
Porting required redevelopment

Data dependency and consistency issues

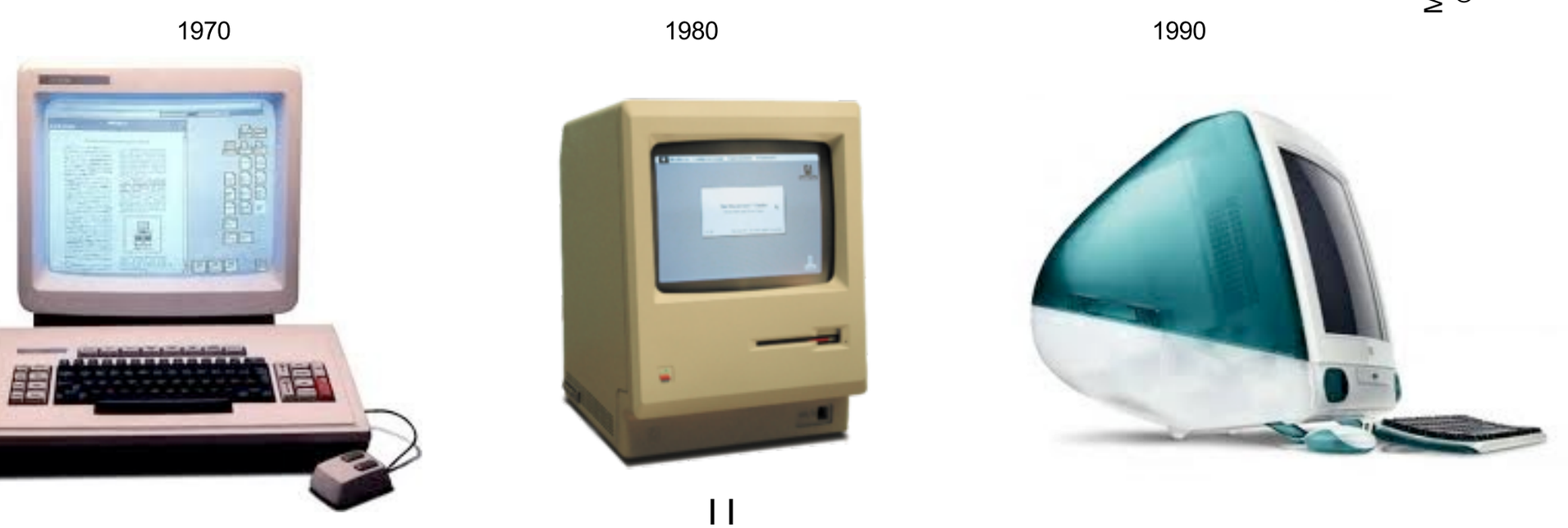




Data management as a primary concern



ease human interaction with the help of knowledge workers



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)

Enterprise Scenario

Early stages

mainframe, assembler language, monolithic applications (including data and textual user interface)

DBMS

application code and (textual, form-based) user interface still entangled

Lowering cost of hw

more separated applications available (different applications in different departments, but hosting related data: redundancy, dependencies)

Changes

Changes were hard to implement!

Hard to track data dependency and replication

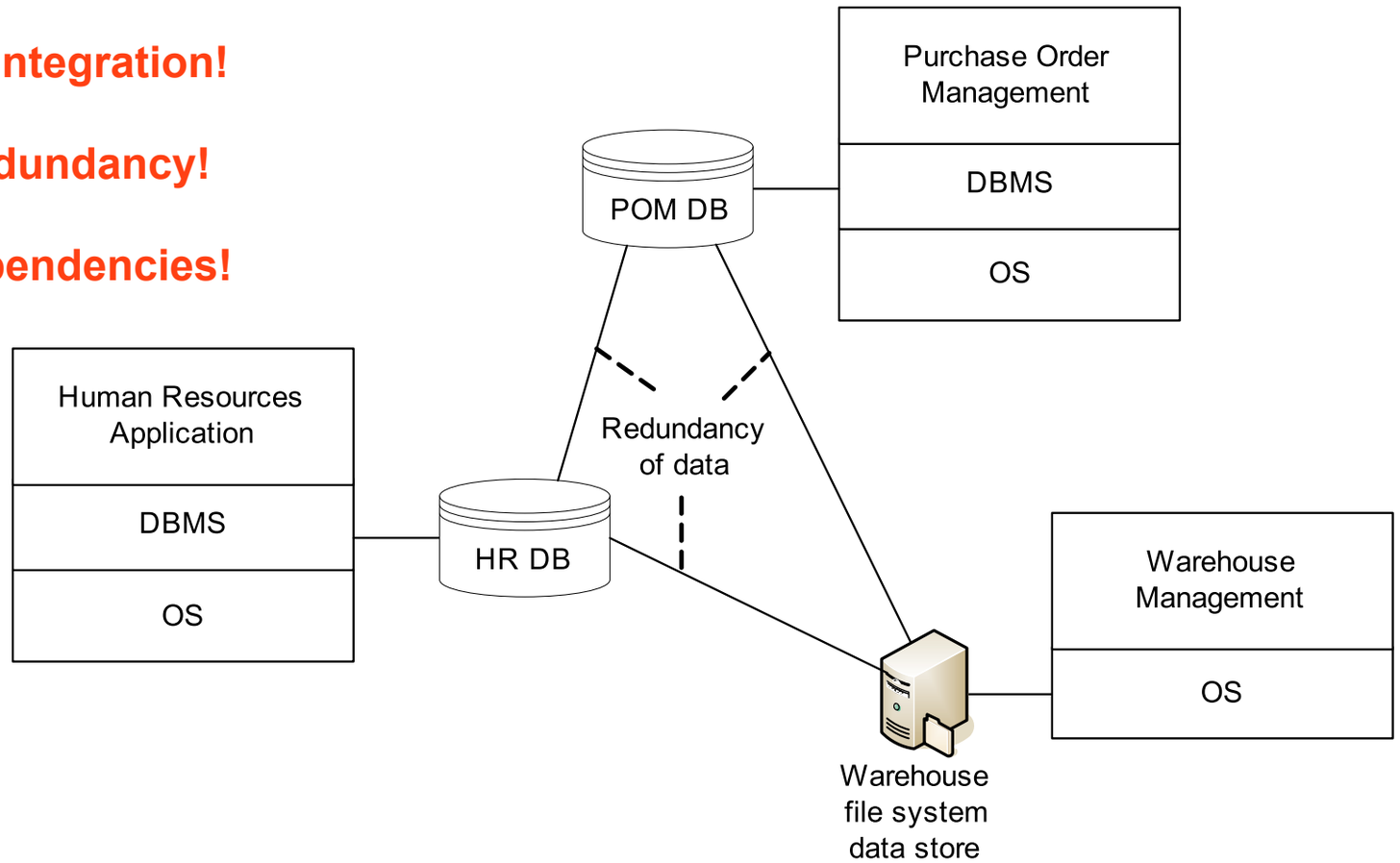
Any modification of an application was a complex and error-prone activity, with domino effect
(e.g. change of customer address format)

Individual enterprise application

Lack of Integration!

Data redundancy!

Data dependencies!



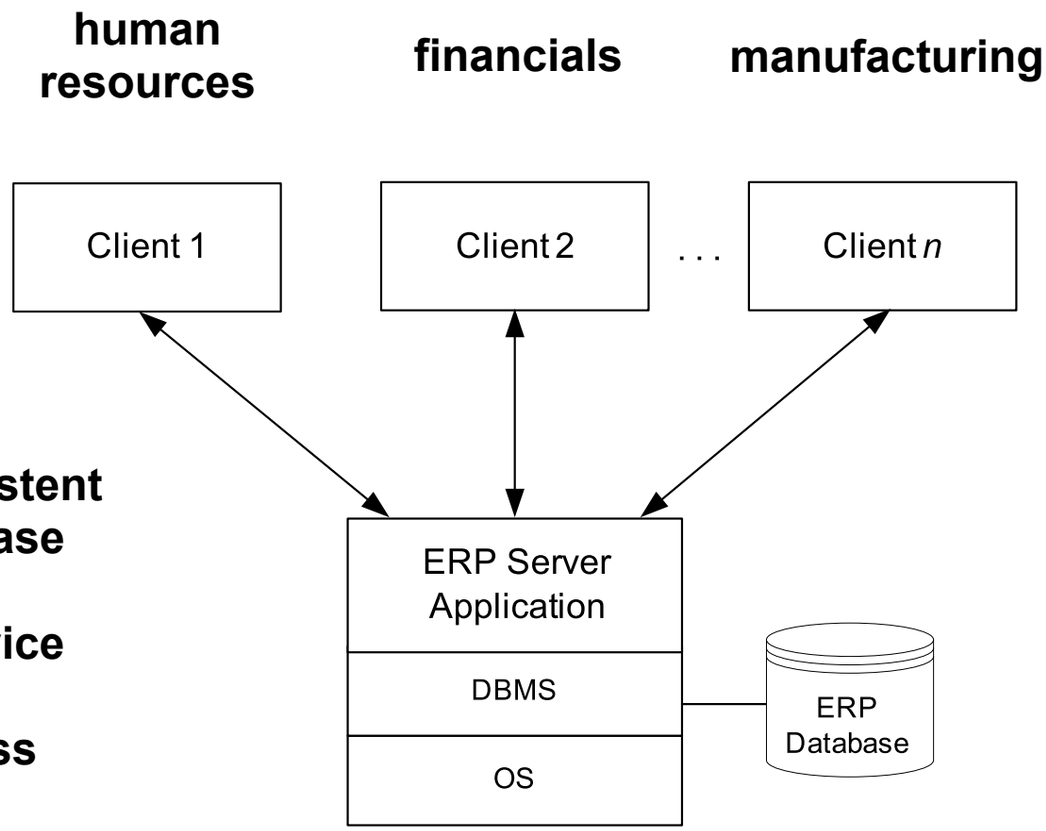
ERP

Enterprise Resource Planning 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

CRM and SCM

New types of sw entered the market around 2000

Customer Relationship Management systems
Supply Chain Management 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

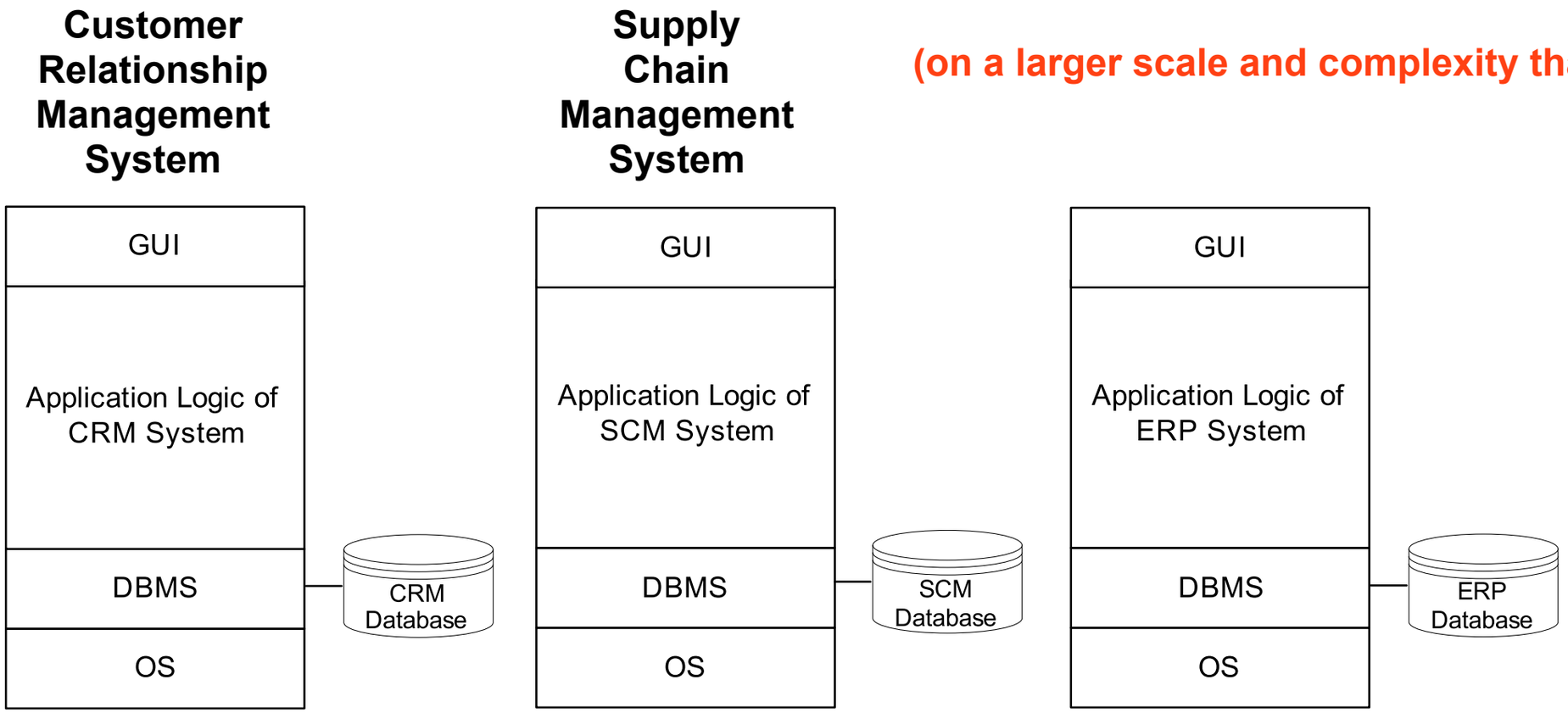
Lack of Integration!

Data redundancy!

Data dependencies!

(on a larger scale and complexity than before)

Data Integration would provide valuable information



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

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

Integration

Manual integration is possible, but:

it consumes considerable resources

it is error-prone

cannot 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

Heterogeneity

Heterogeneous information technology landscape
has grown in an evolutionary way for years:
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 CAstreet),
fields with the same name but different meaning
(e.g. Price, with or without taxes?, unitary?)

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.

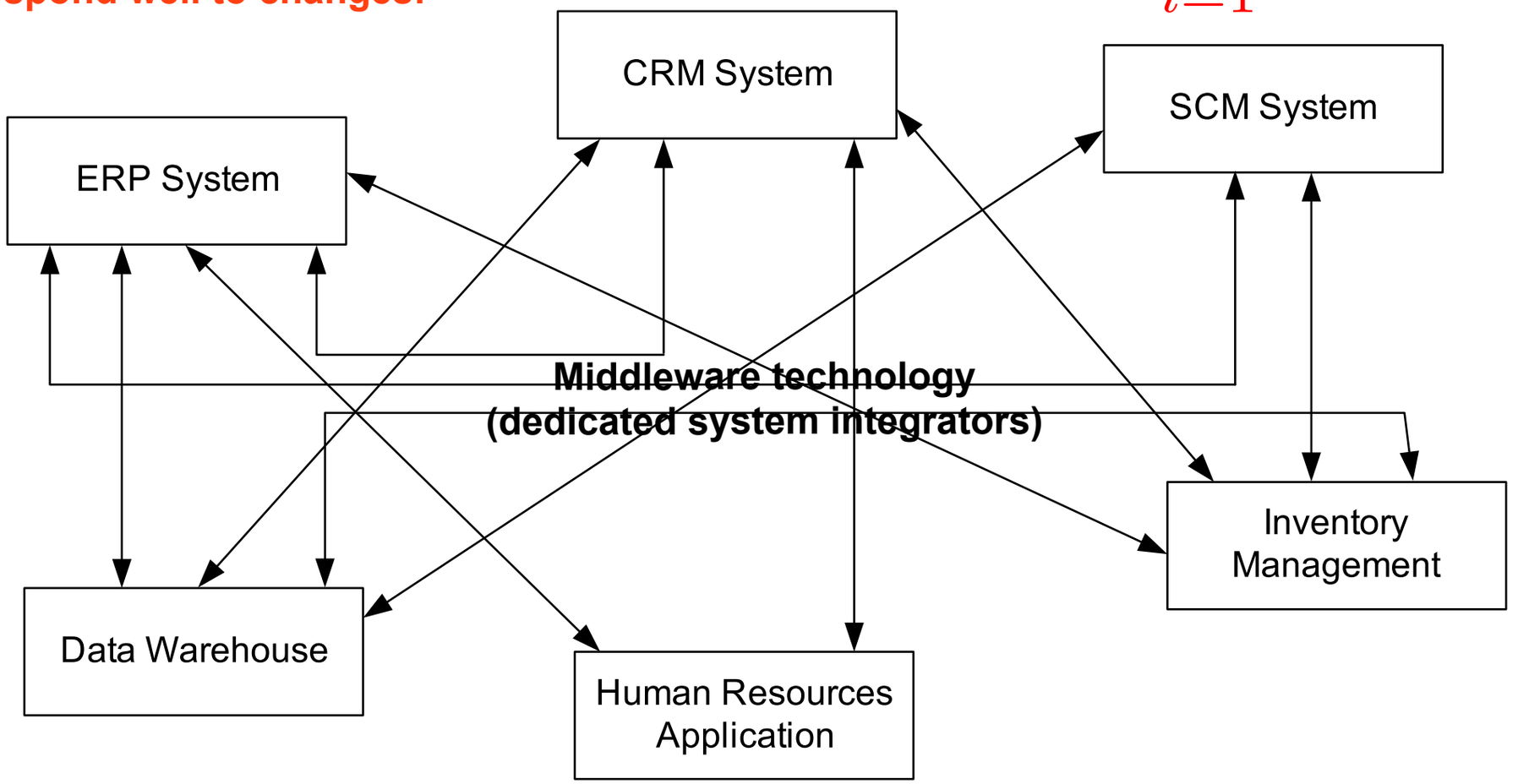
Point-to-point integration (of silos)

N x N hard-wiring problem!

Too many interfaces to develop!

Does not respond well to changes!

$$\sum_{i=1}^{N-1} i = \frac{N(N-1)}{2}$$



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

Support Changes, efficiently, effectively

The point-to-point approach opposes some resistance to fluent changes

Hard-wiring of interfaces (and their numbers) is the main limit

Reprogramming an interface requires considerable resources, typically

Alternative

Move to message-oriented middleware

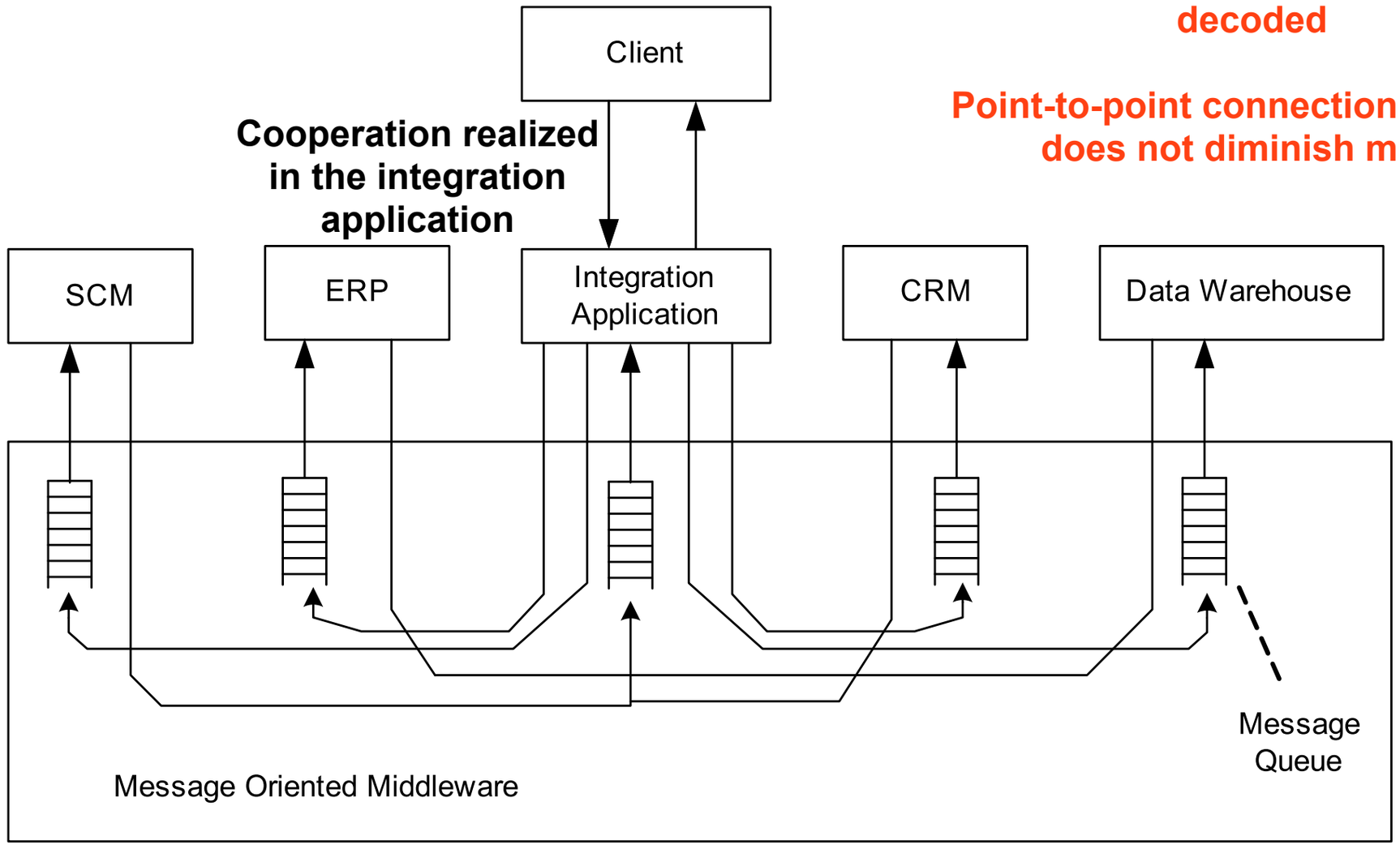
Message Oriented Middleware

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



Messages must be encoded and decoded

Point-to-point connection problem does not diminish much

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

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

Hub-and-Spoke

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 represents the spokes

Interactions between any two application must pass through the hub

Important feature
Sender of a message does not need to encode the receiver of the message

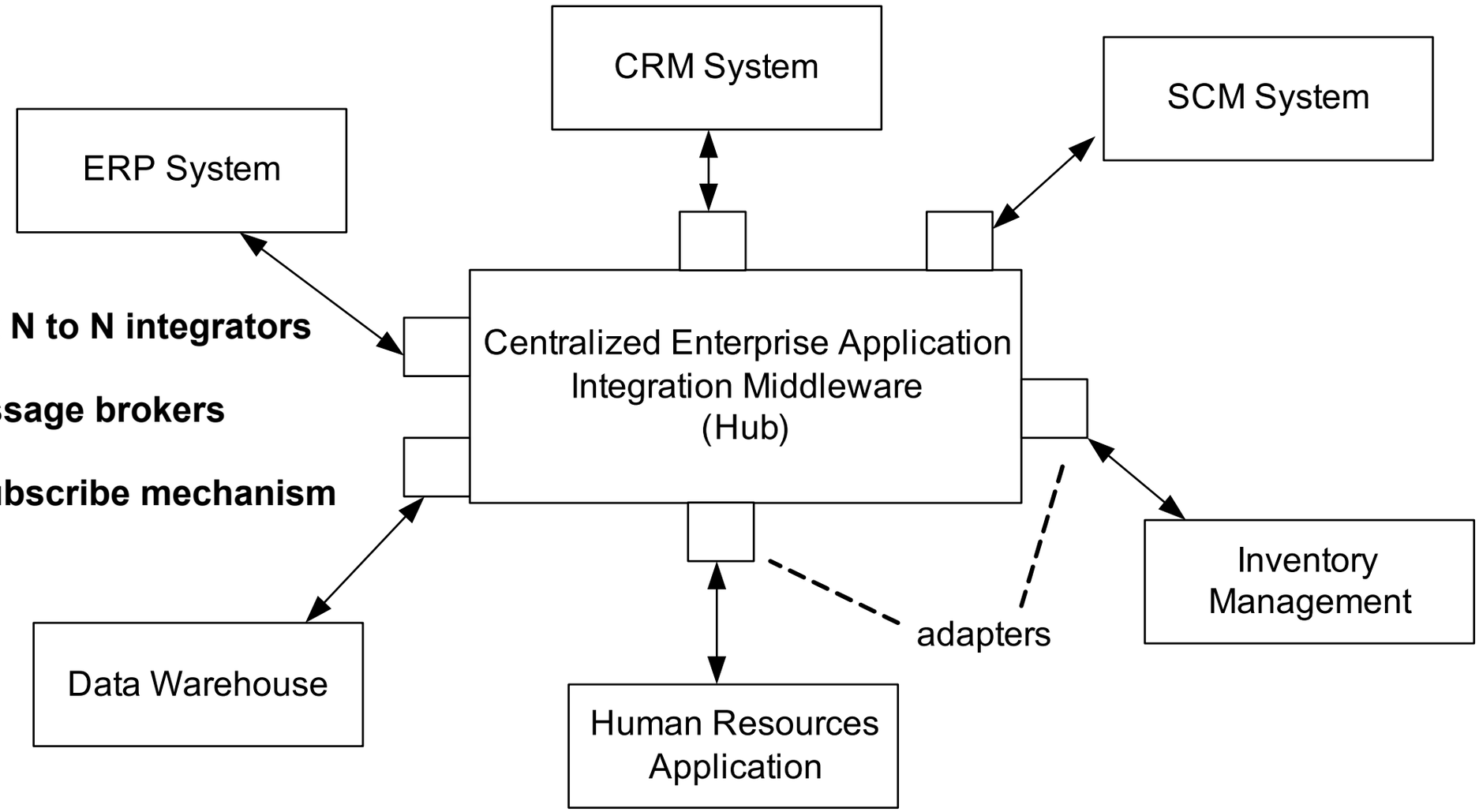
Hub-and-spoke integration

Configuration and management of adapters and message brokers can become cumbersome

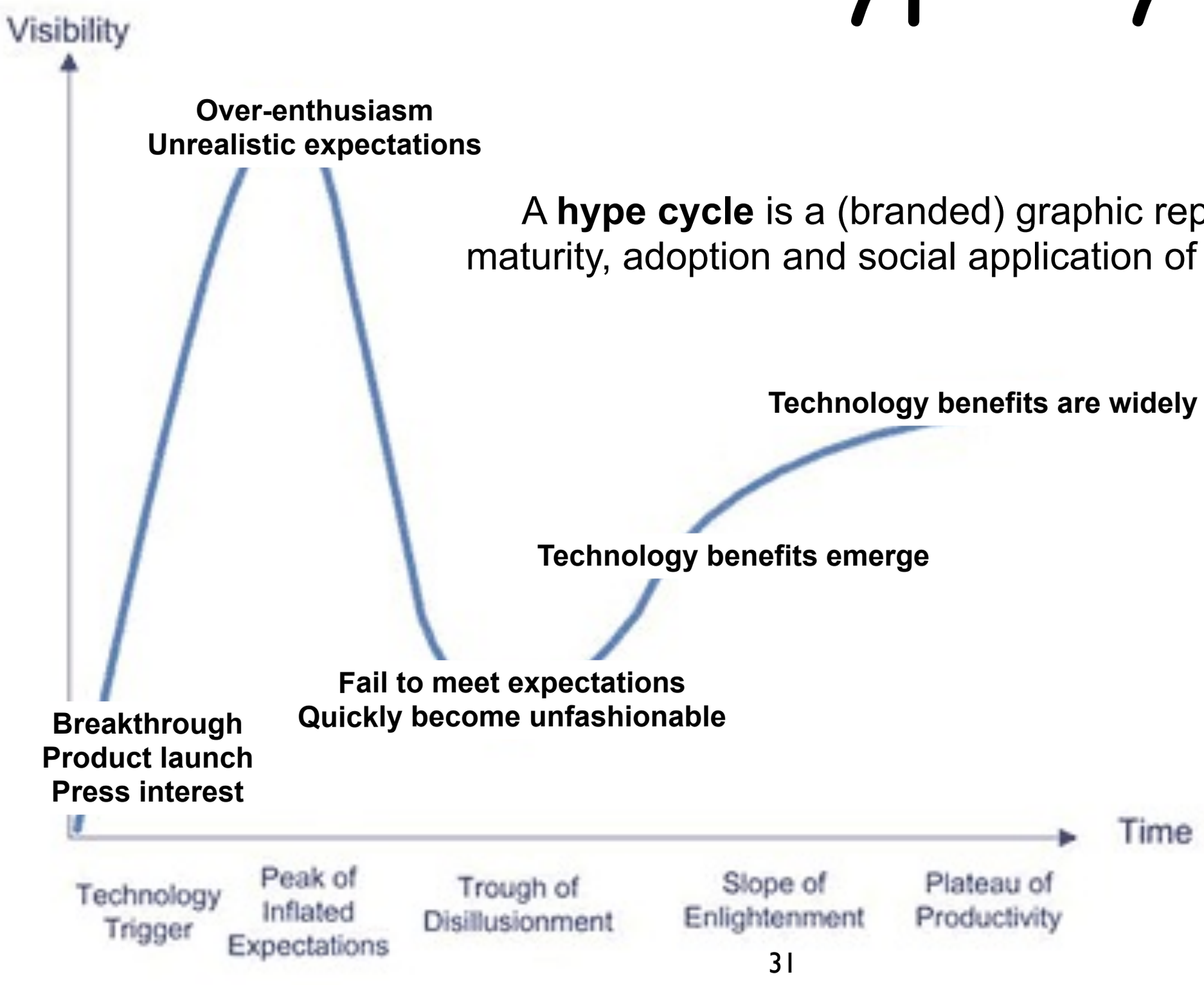
From N x N to N integrators

Message brokers

Publish/subscribe mechanism



Gartner's hype cycle



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

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

Building interfaces is an art

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

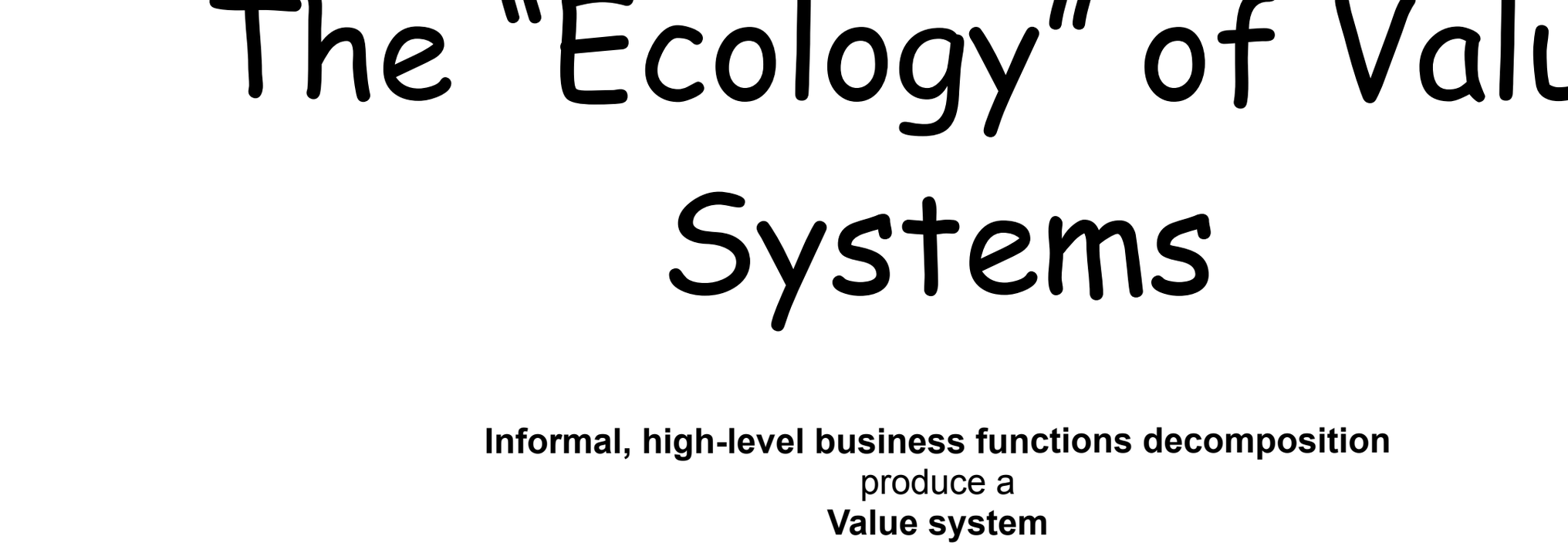
as a means to functionally break down the activities a company performs and to analyze their contribution to the commercial success of the company

Process orientation

as the way to organize the activities of enterprises

The "Ecology" of Value Systems

Informal, high-level business functions decomposition
produce a
Value system
made of
Value chains
centred at the enterprise E under consideration



Process Orientation

The mid 90's saw process orientation as a strong development not only to capture the activities a company performs, but also to **study** and **improve** the **relationships** between activities

Business process reengineering is based on the understanding that the products a company offers to the market are provided through **business processes**, and that **rapid, radical redesign of these processes is the road to success**

Taylorism

Process orientation is based on a critical analysis of a concept to organize work units originally introduced by Frederick Taylor to improve industrial efficiency

Taylorism uses functional breakdown of complex work to small granularities

Then, highly specialized work force can efficiently conduct these work units of small granularitiy

Taylorism has proved very successful in manufacturing and fuelled the industrial revolution

Handovers

Fine-grained activities require many handovers of work in order to process a given task

Until early nineteenth century the products were typically assembled in a few steps only, so handovers were not introducing much delays

Moreover, tasks were of simple nature and did not require any context information on previously conducted steps

Taylorism proved inefficient for organizing work in modern enterprises

Pitfall of Taylorism

Steps of a business process are often related to each other

Context information on the whole case is required during the process

The handovers of work cause a major problem because of that (workers required knowledge)

In the end, functional breakdown proved inefficient in modern business organizations that mainly process information

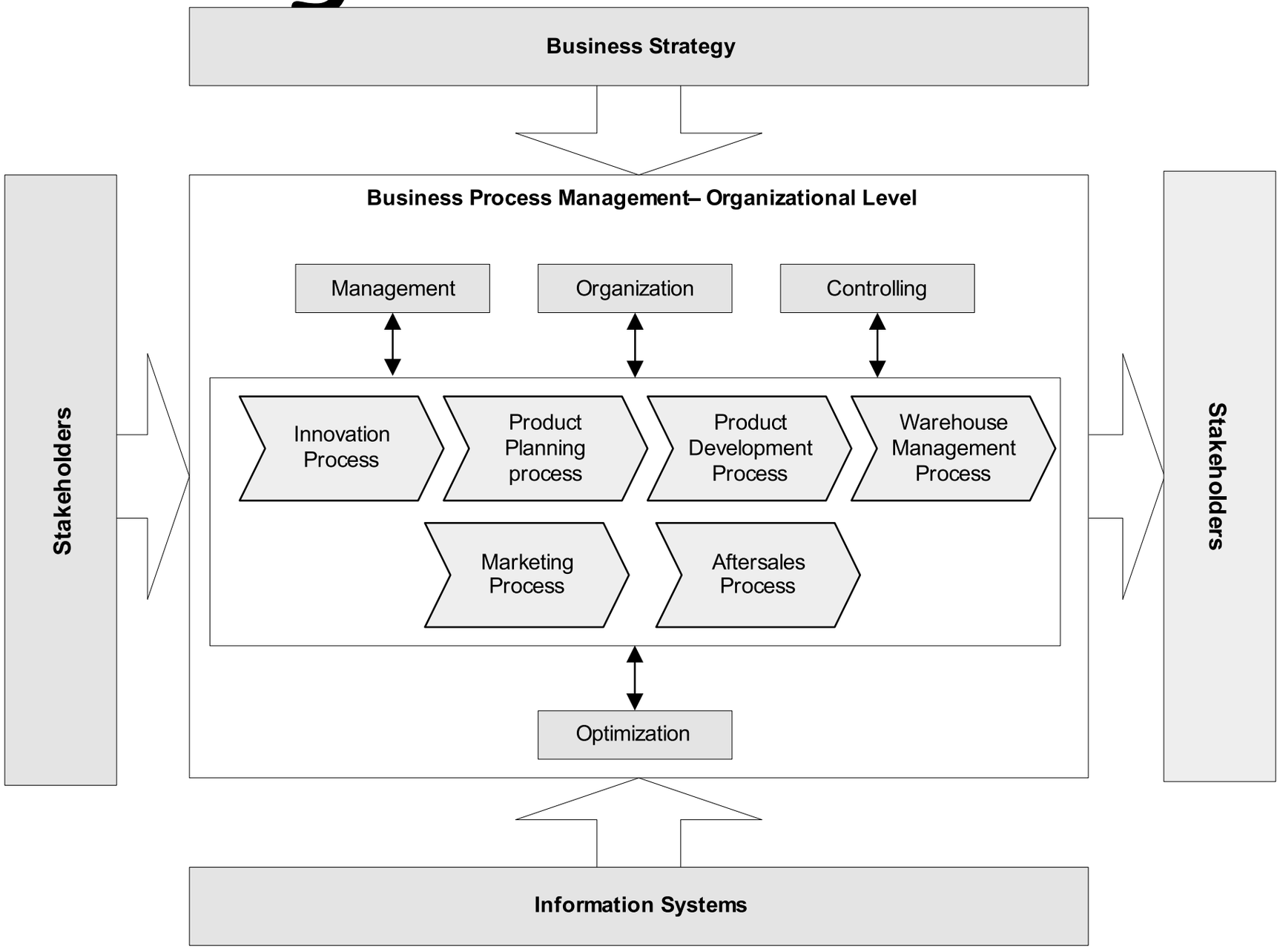
Process perspective

It is instrumental to combine multiple units of work of small granularity into work units of larger granularity to reduce the handover of work

As a consequence, workers must have broader skills and compentencies
(knowledge workers must have a broad understanding of the ultimate goal of their work)

Main effect, at the organizational level, process orientation led to the characterization of high-level operations (usually, less than a dozen), called **organizational business processes**

Structure of High-Level Organizational BP



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

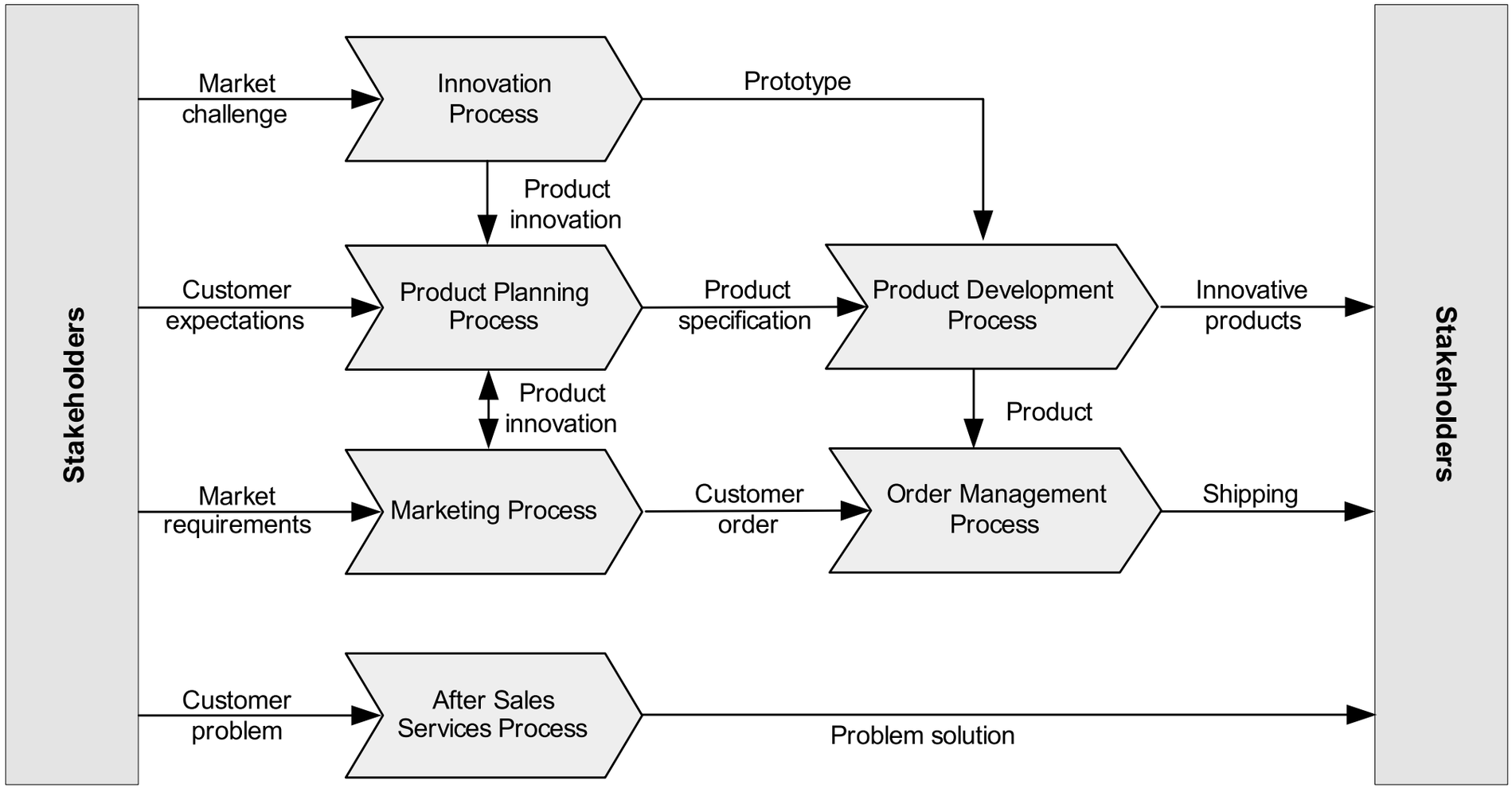
Concepts from business administration

Top-level: Form-based description of organizational business process (black-box view)

Process Name: Product Development Process	**Responsible Process Manager:** Dr. Myers
From: Requirements **To:** Rollout	**Type:** Development Project
Process Inputs: Requirements Document, Project Plan, Budget Plan, Prototyps	**Supplier Processes:** Product Planning Process, Innovation Process
Process Results: Integrated and completely tested innovative product with complete documentation	**Customer Processes:** Order Management Process, After-Sales Service Process

Concepts from business administration

Lower-level: dependencies are shown in a process landscape



Workflow management

Needs of:

Explicit representation of process structures in process models

Controlled enactment of business processes according to these models

Workflow re-birth

Born as rational organization of work in manufacturing:
optimization of throughput and resource utilization

Re-born in ICT:
flexibility, adaptability, modularity, distribution

Workflow management coalition (WfMC)

Founded in the ‘90s by vendors, users, academia:

http://www.wfmc.org

The model-driven approach facilitates a high degree of flexibility:

old process models can be adapted to fulfil new requirements and the new processes can be readily enacted

Workflow

Definition: a **workflow** is the automation of a business process, in whole or in part,

during which documents, information, or tasks are passed from one participant to another for action,

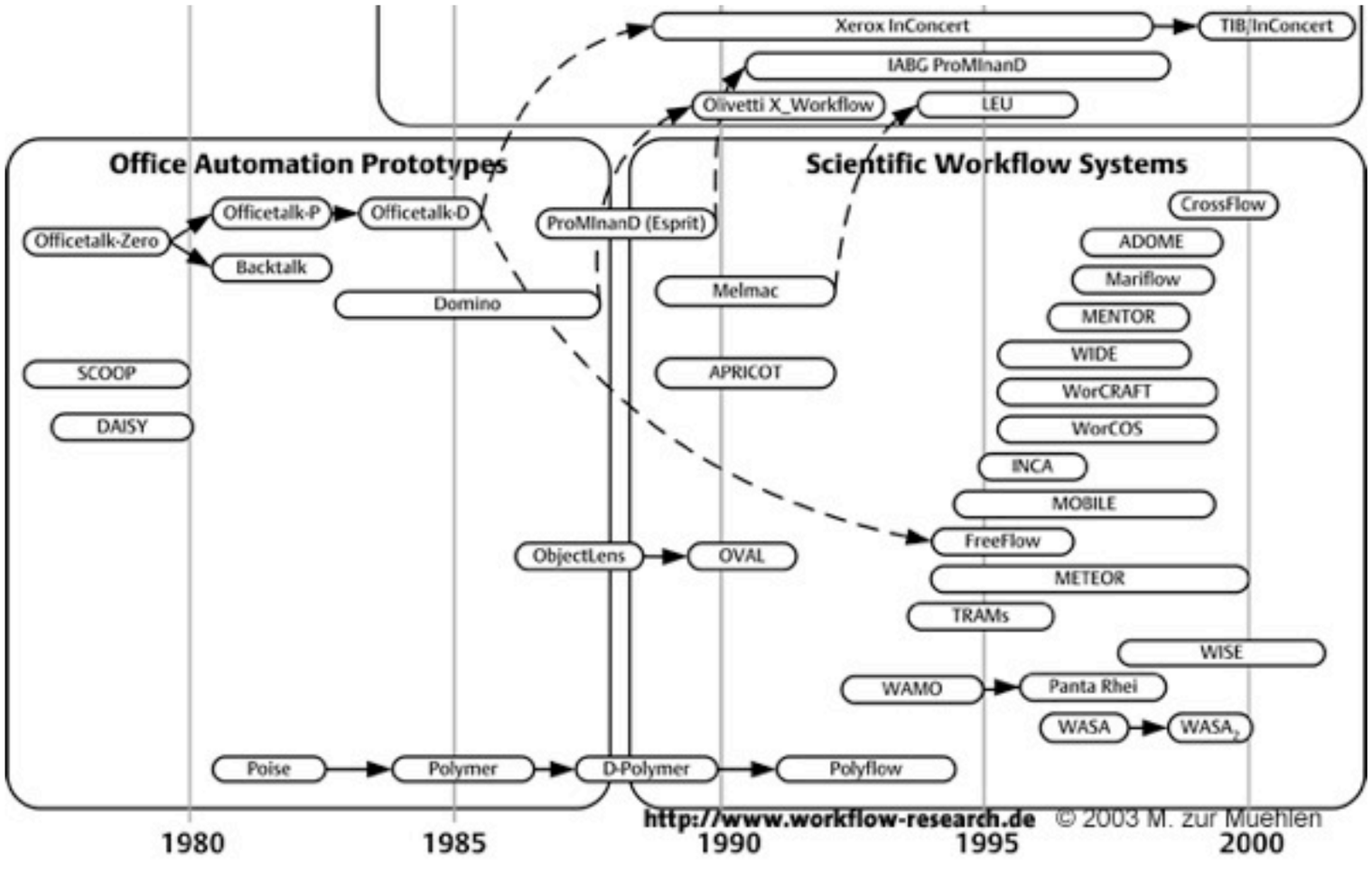
according to a set of procedural rules

Workflow management system

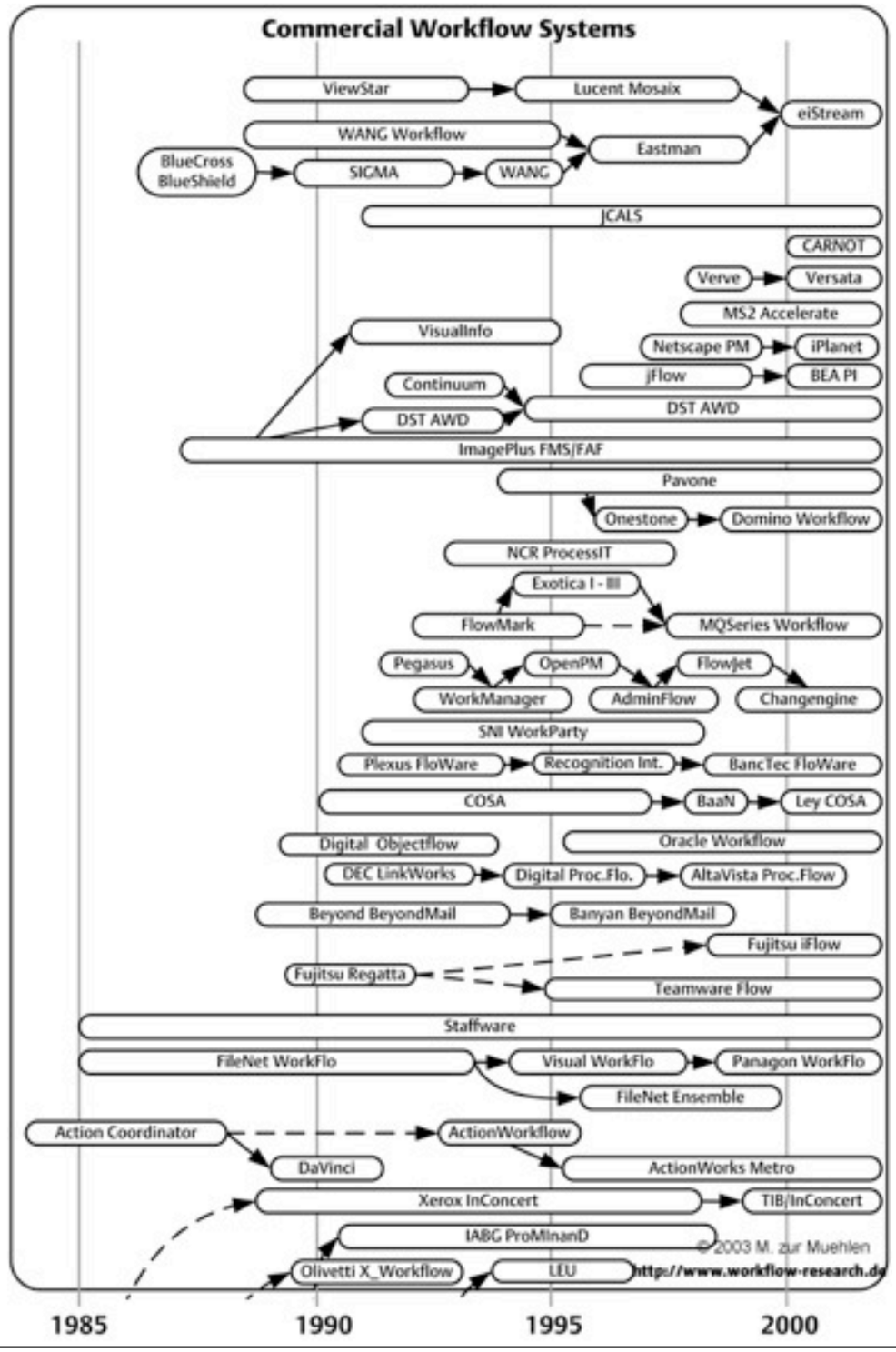
Definition: a **workflow management system** is a software system
that defines, creates, and manages the execution of workflows through the use of software, running on one or more workflow engines,
which is able to interpret the process definition, interact with workflow participants, and,
where required, invoke the use of IT tools and applications

A piece of history

Research

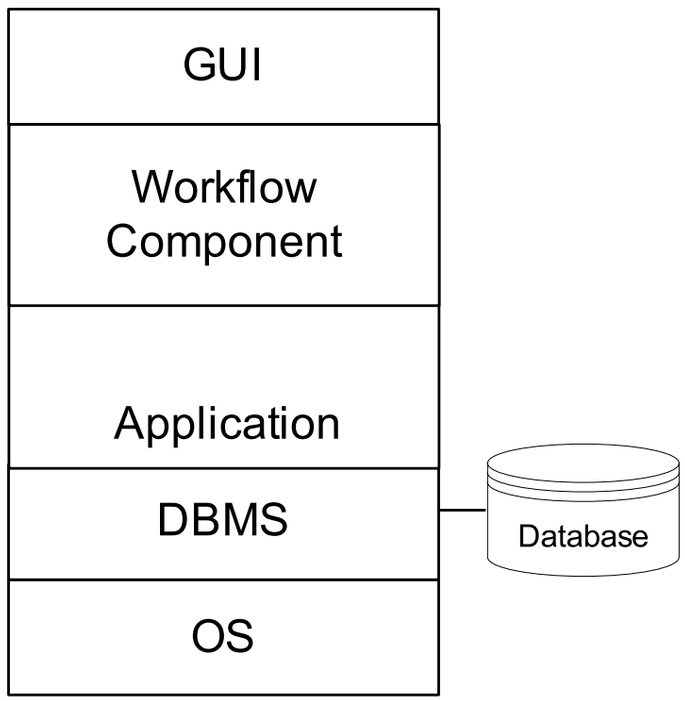


Systems



Workflow component

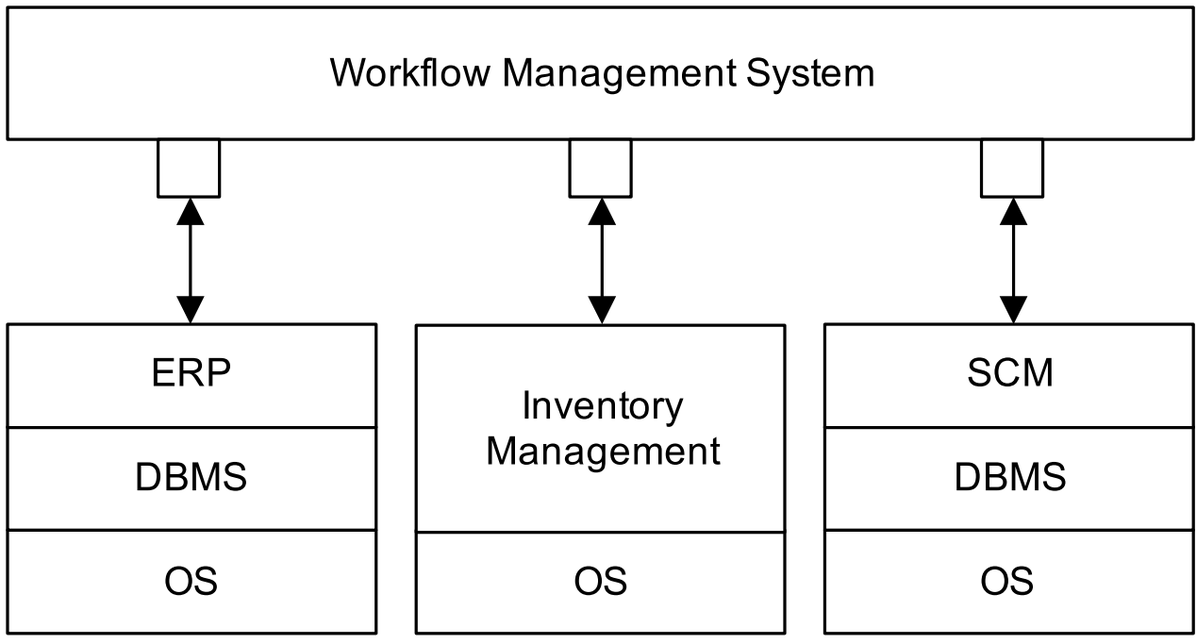
Definition: a **single-application workflow** consists of activities and their causal and temporal ordering that are realized by one common application system.



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

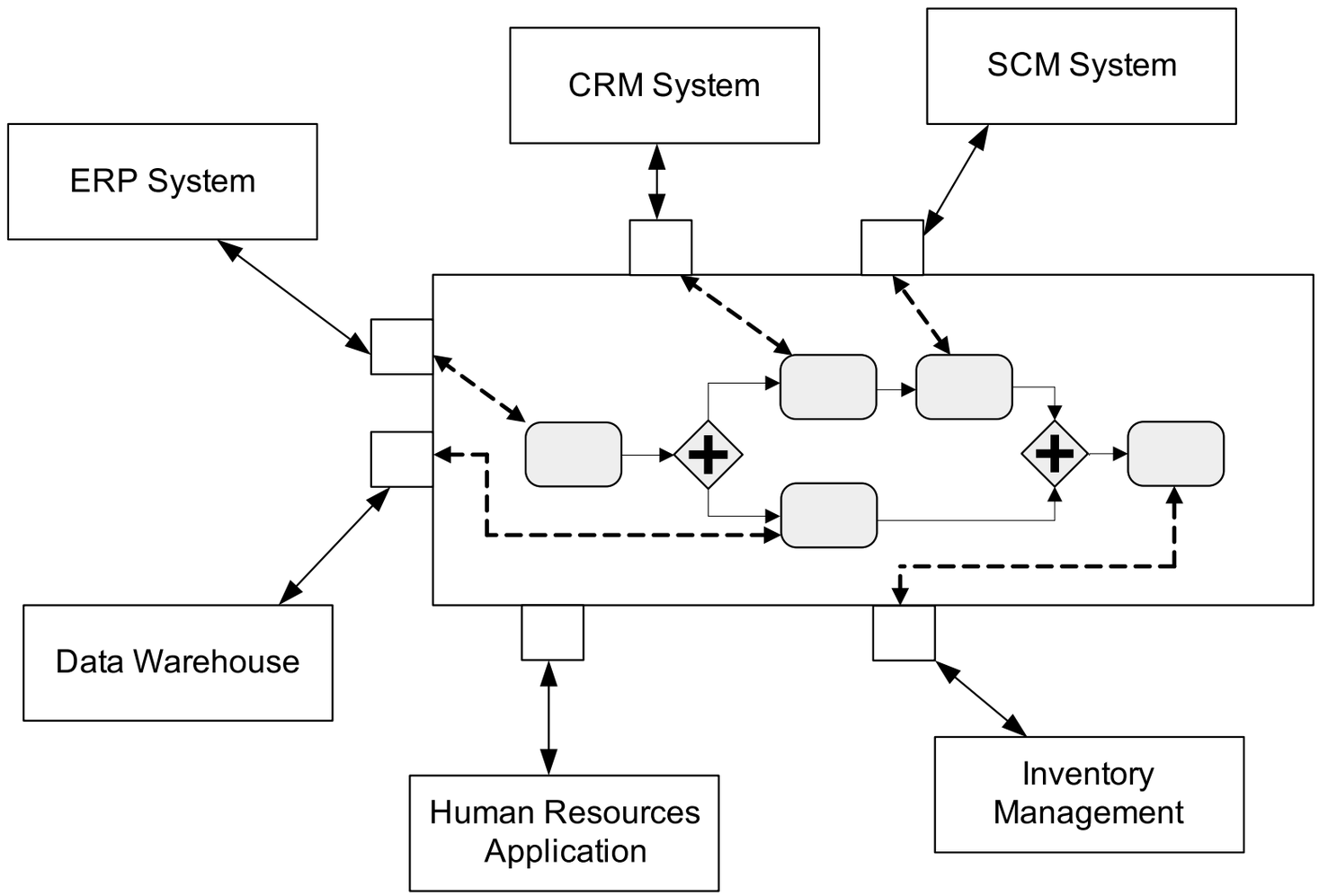
Multiple-application workflow system

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



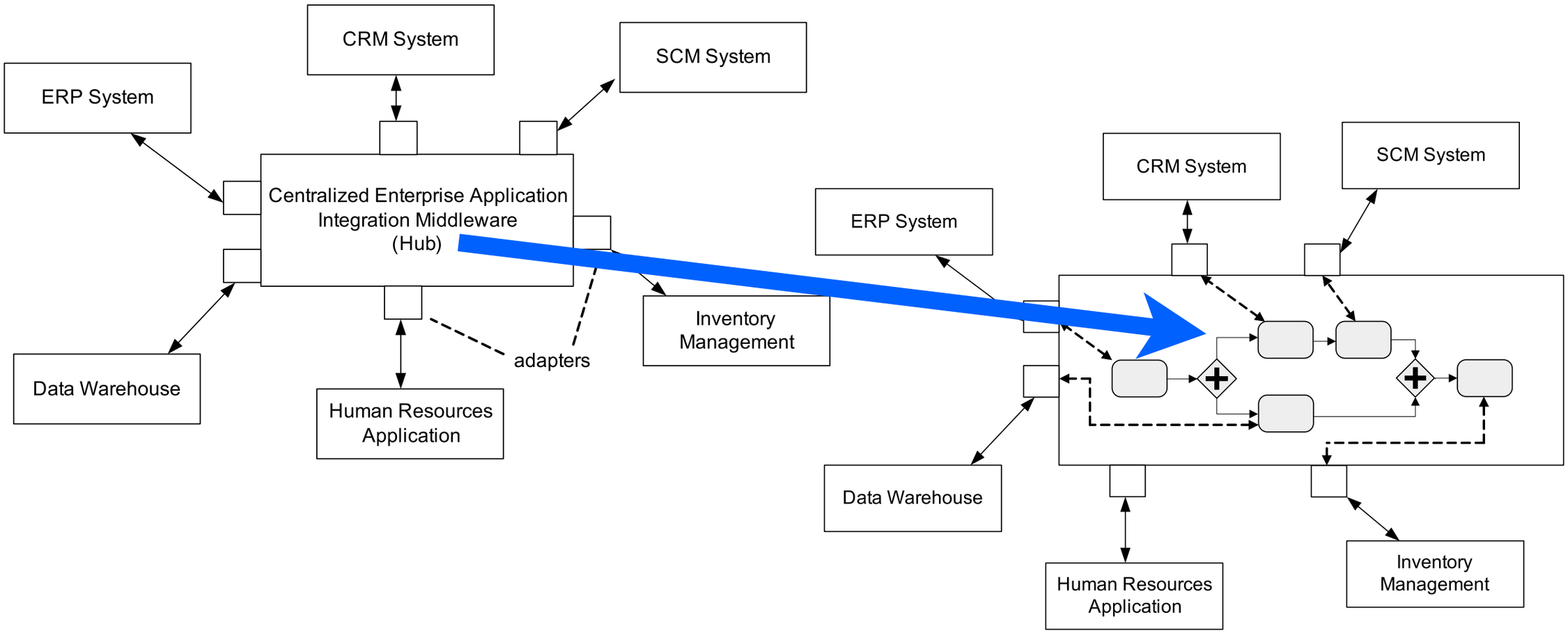
System workflow

Definition: a **system workflow** consists of activities that are implemented by software systems without any user involvement.



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

Do you remind hub-and-spokes EAI?



Limitations in workflow management

Technical integration problems:

Scarcely documented applications

Different levels of granularity

Tight coupling of applications
(direct invocation)

Office automation system

When task performed by humans are involved in the workflow, it is not sufficient to equip workers with adequate software:

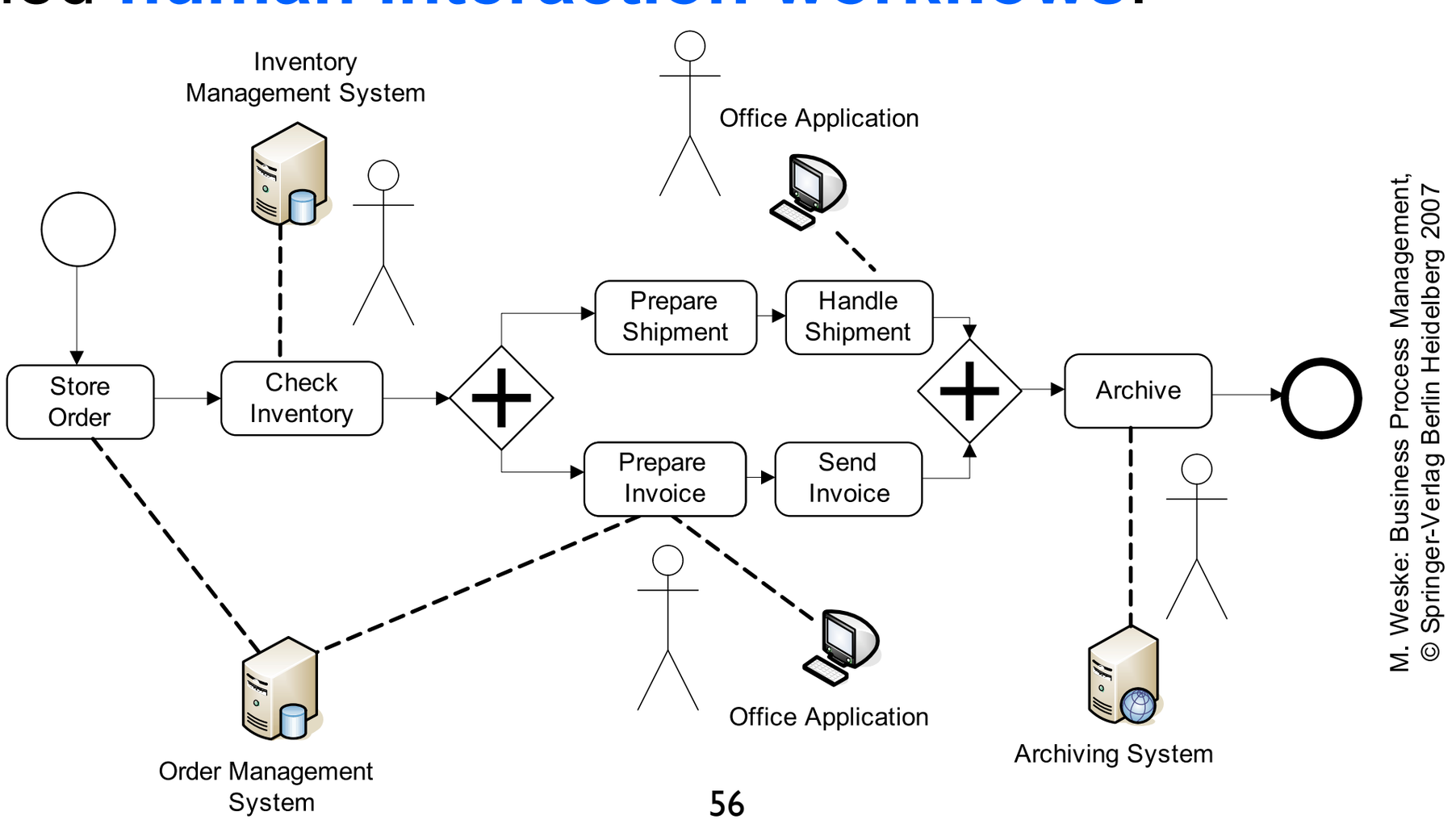
their collaboration must be supported

shared data repositories and work handover can speed-up office procedure considerably

First attempt: office automation
(form-based applications, quite narrow)

Human interaction workflow

Definition: Workflows in which humans are actively involved and interact with information systems are called **human interaction workflows**.



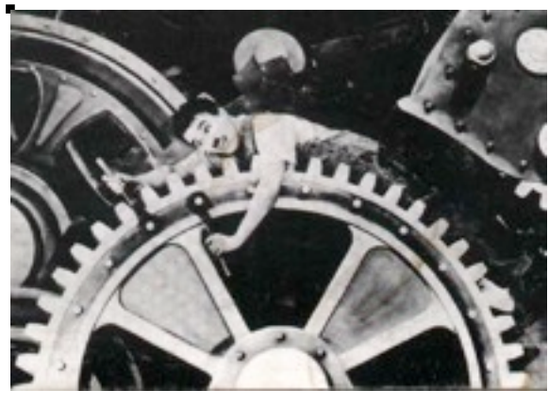
Concepts in human interaction workflow

Roles = groups of employees that qualify for being responsible of certain activities

Work item list
(also called in-basket)

Limitations in workflow management

Problems with knowledge workers:



User acceptance issues

Machine burdening of workers

Little room for creativity

Enterprise service computing

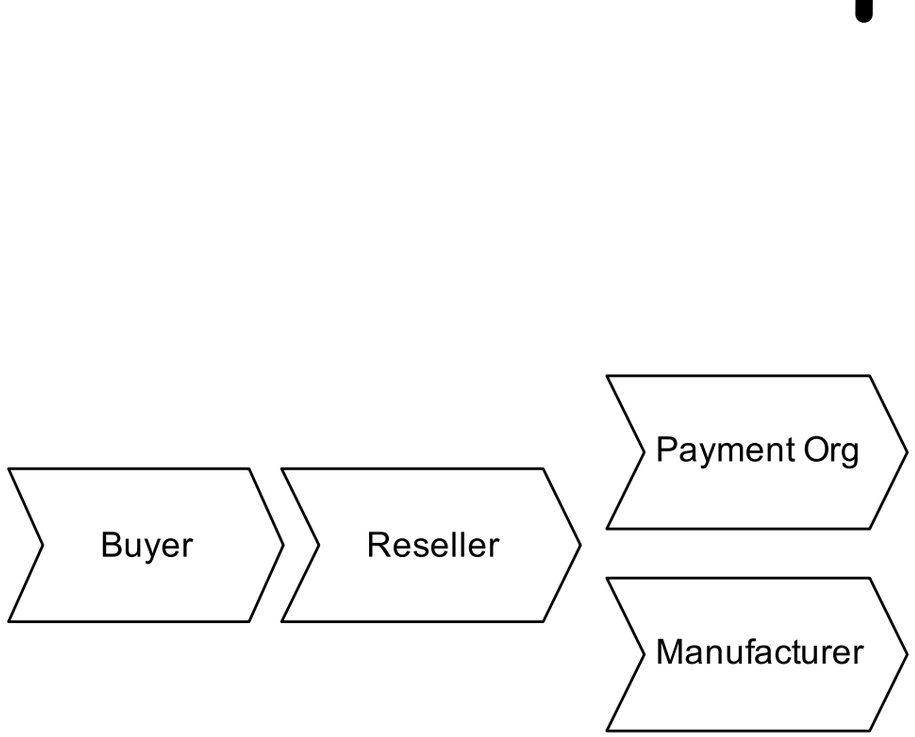
Main idea:

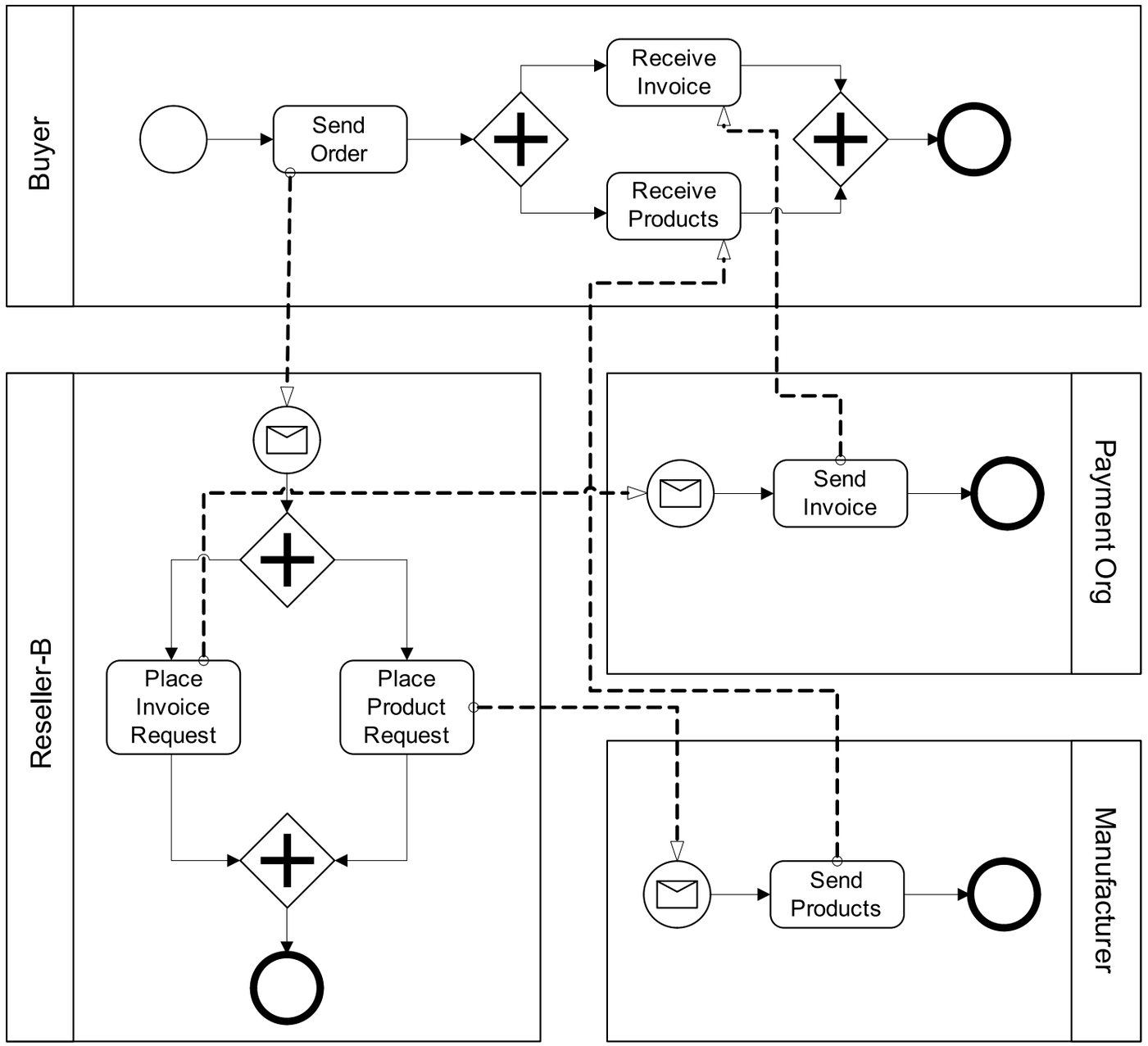
Business functionalities exposed as services

Services are equipped with usage information

Customers can find services and use them

Business-to-business processes





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**.

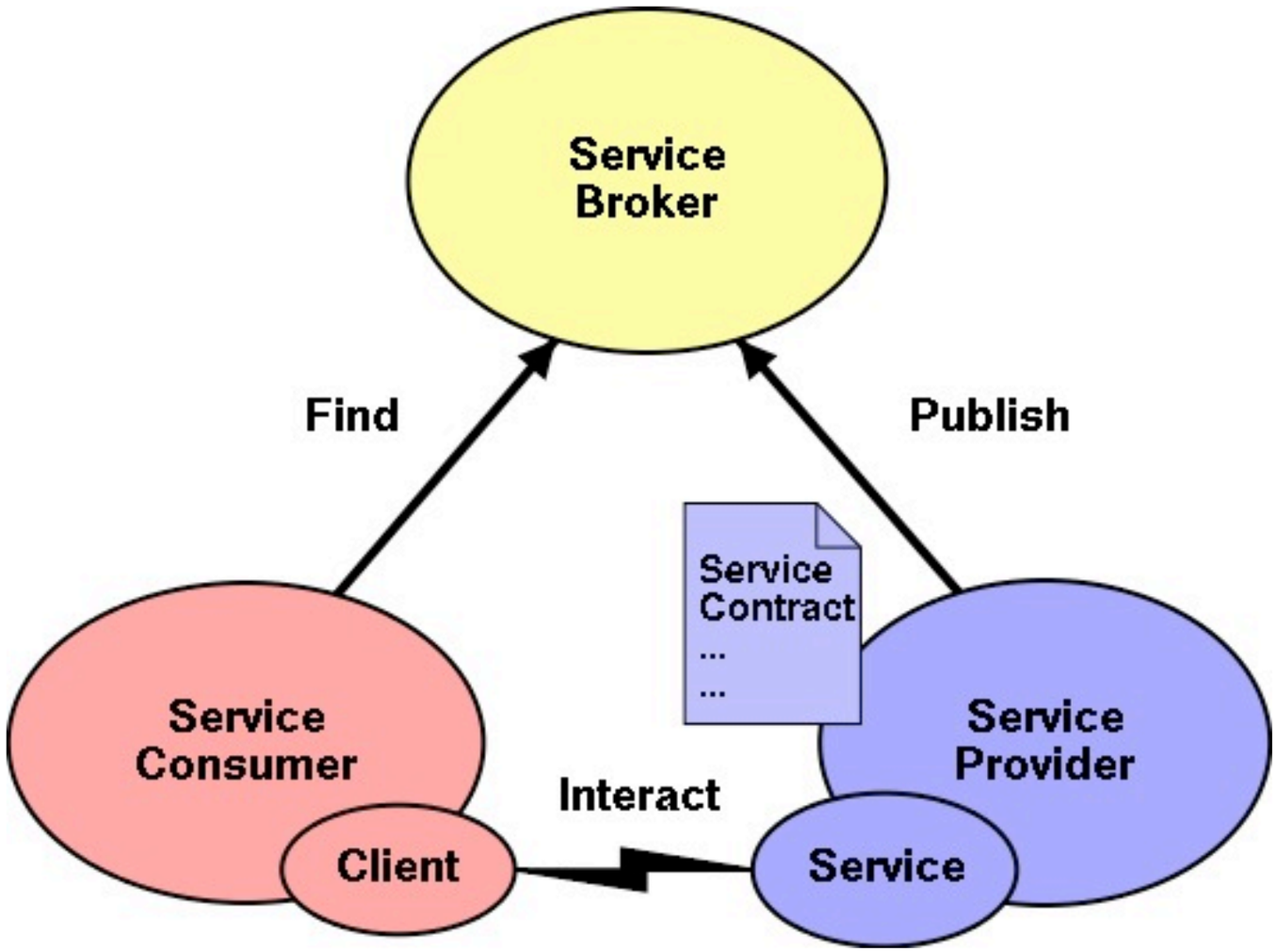
Service descriptions provide a level of detail that facilitates service requestors to **bind** and **invoke** them.

Service-oriented architectures

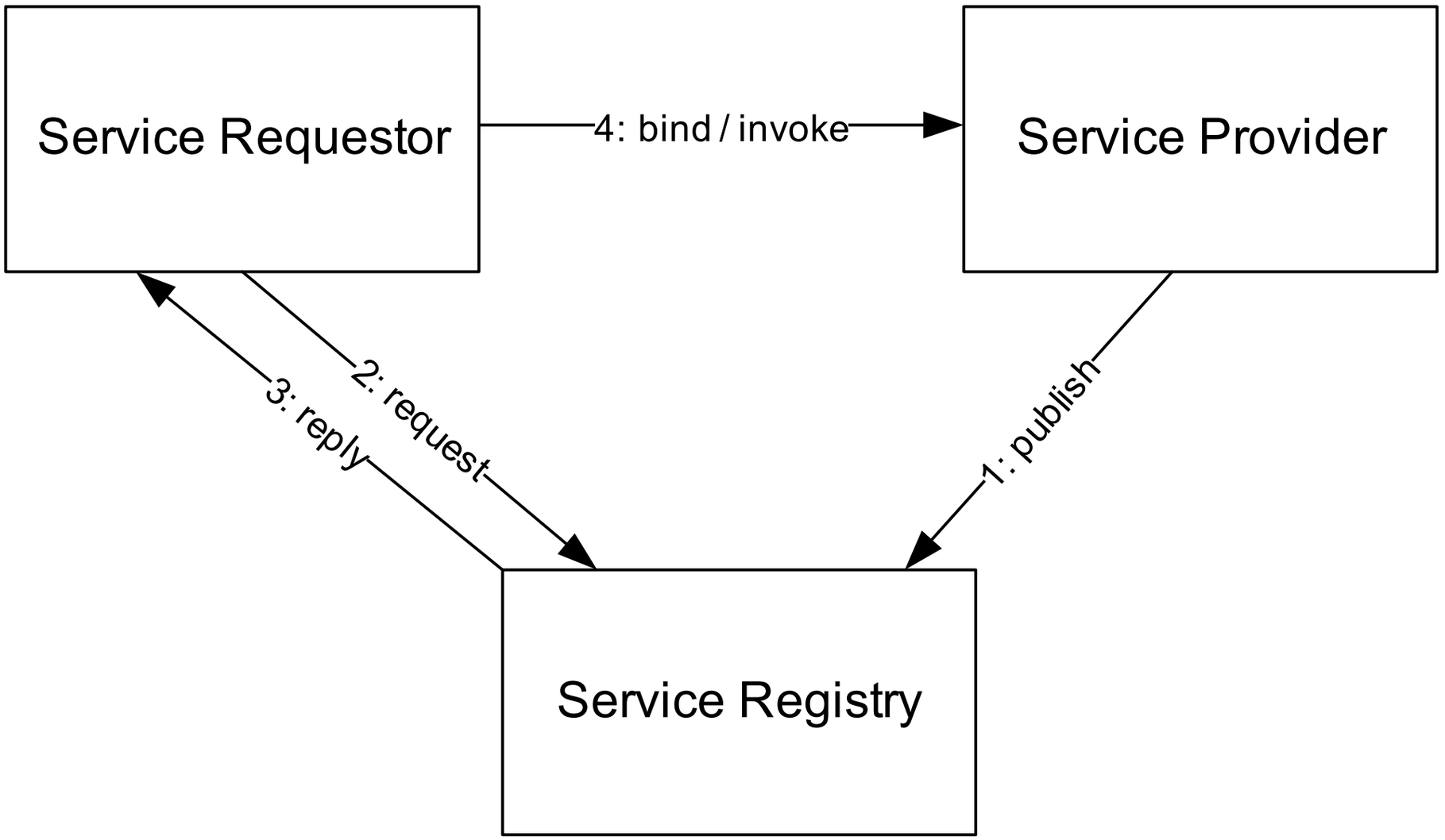
To work easily, flexibly, and well together, services must be based on shared organizing principles that constitute a service-oriented architecture.

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-oriented architectures



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

Advantages of SOA

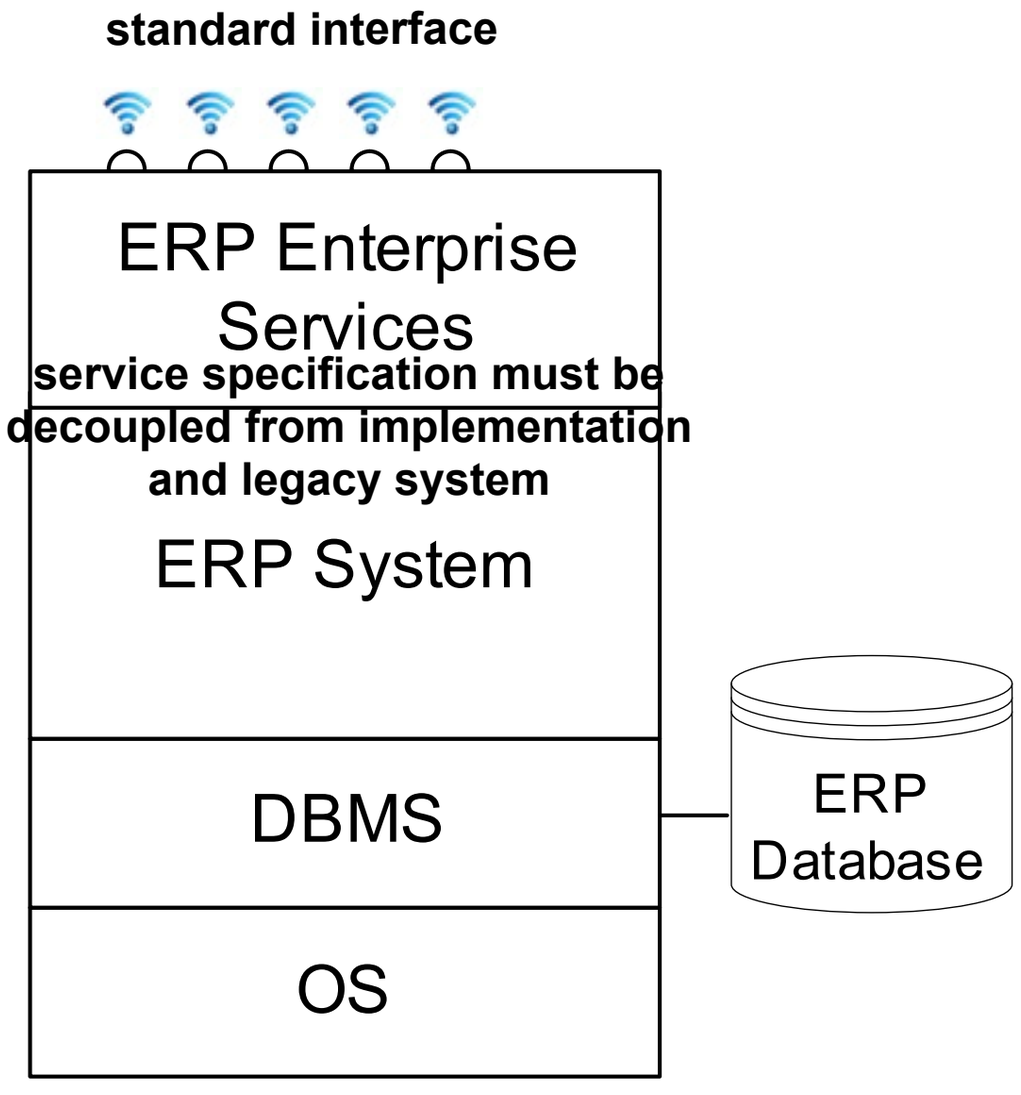
Reuse of functionality at coarse level of granularity

New applications can be built with less effort

Existing applications can be efficiently adapted to changing requirements

Reduced maintenance and development costs

Service enabled application system



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

Products as services

Corporations are increasingly perceived by the set of services they provide

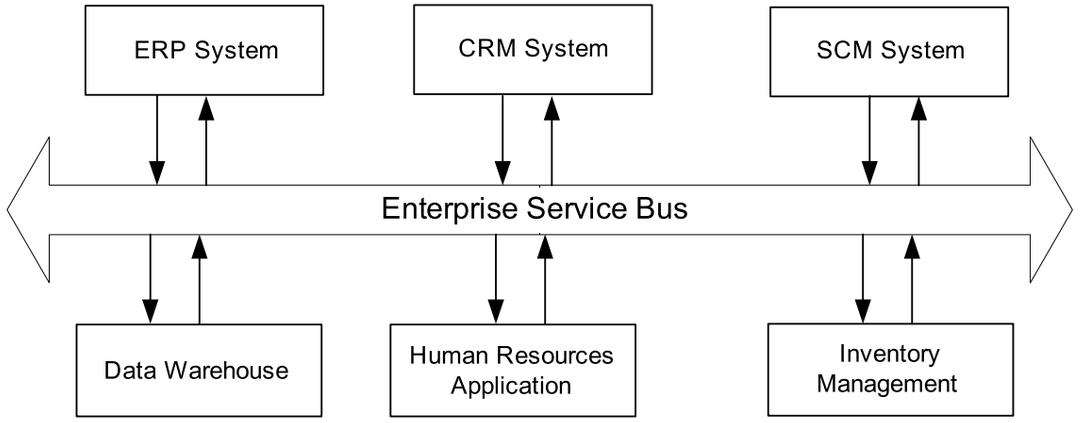
These services exposed to the market can be realized by enterprise services (provided by the back-end application system)

Also services provided by third parties can be integrated so that better end used services can be provided to the customer

Enterprise service bus

Centralized component that integrates all applications

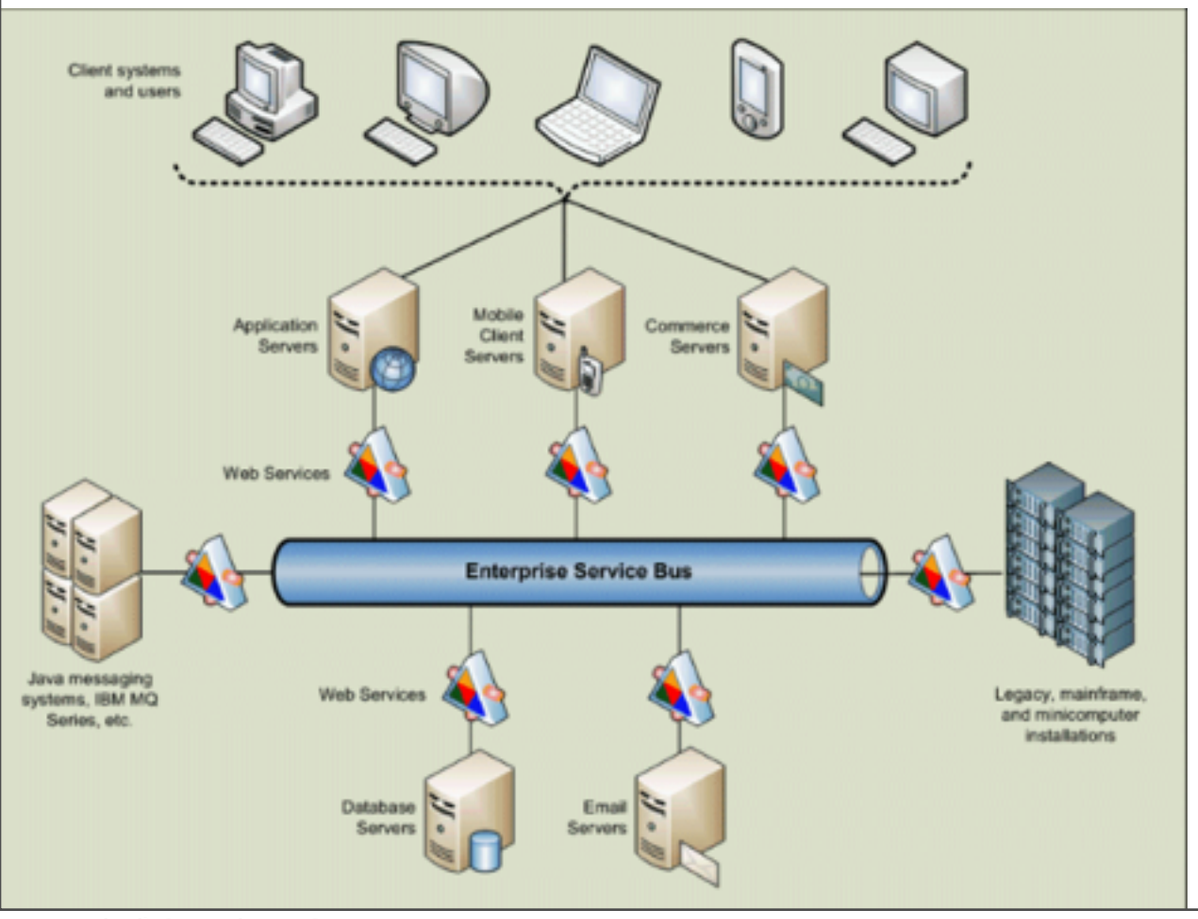
Hides heterogeneity by introducing service interfaces

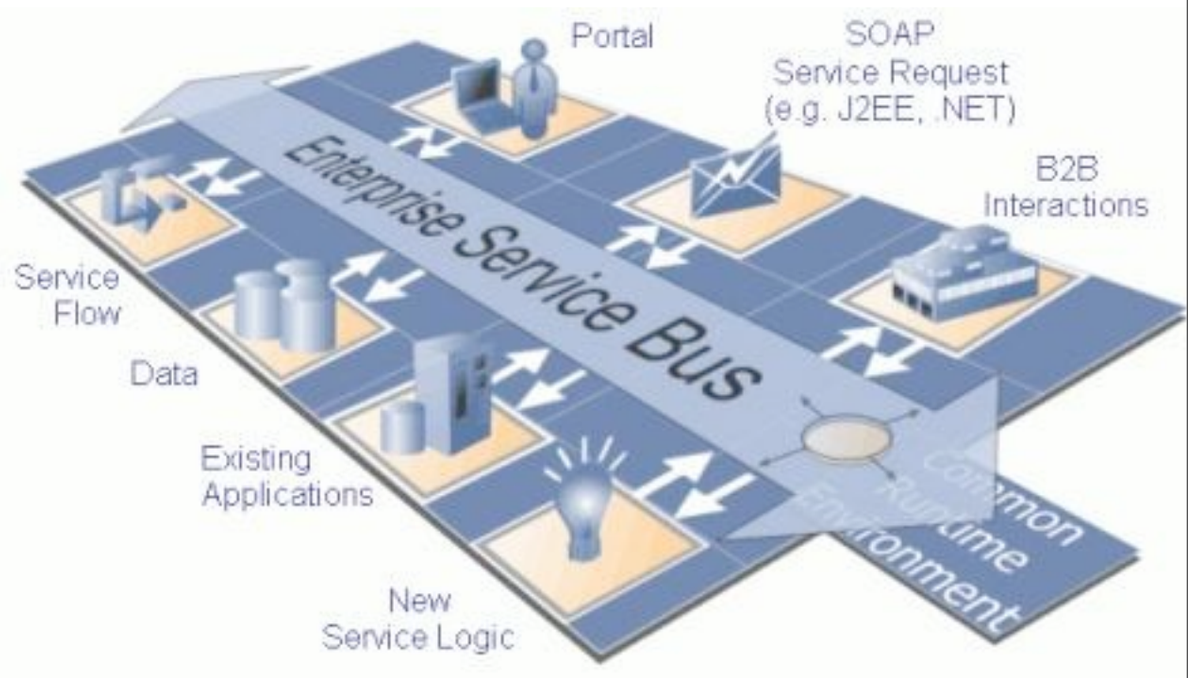


M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2007

Local registry

Manual search (absence of dynamic matchmaking)

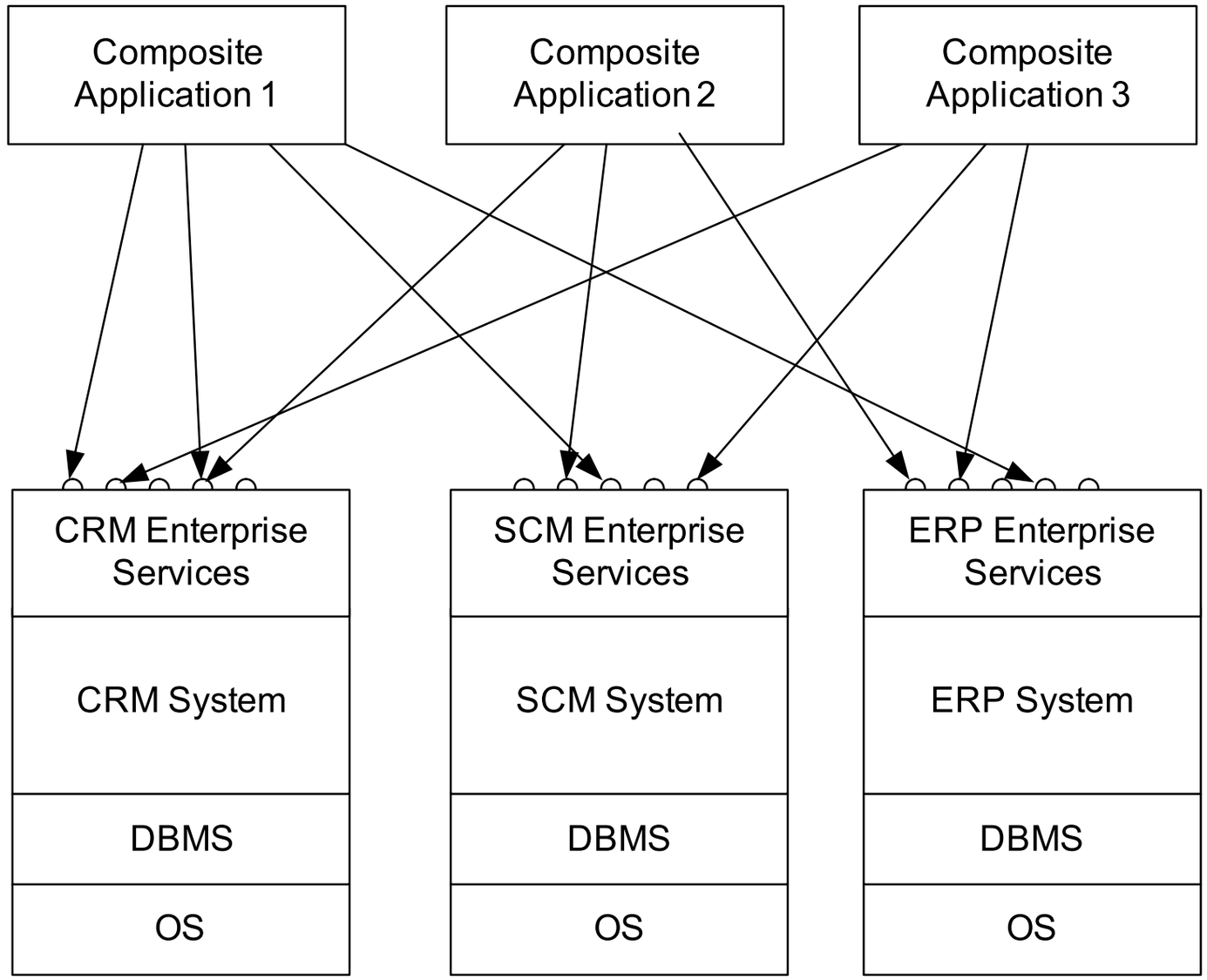




Composite service based application

Intra-company

well-expressed as business processes



Local registry

Manual search (absence of dynamic matchmaking)

The big picture