Architecture & UML analysis

How to develop your SOA to empower your IT strategy ?

Frédéric Vermaut,
Hermès IT Architecture Director

Who am I ?

- Frédéric Vermaut
- 41 years
- IT Architecture Director
- Architecture & Methodology specialist
- Project Manager
- IT Experience since 17 years
  - Ducroire, ING, Fortis, TUC Rail, AXA, Dexia…
- Civil Engineer

What is Architecture ?

What is Architecture ?

- What is IT architecture ?
  - « set of significant decisions about the organization of a software system »(Booch, Rumbaugh, Jacobson)
  - « Set of statements describing software components and assigning the functionality of the system to them »
  - "Breakdown of a system into its parts & decisions that are hard to change" (Fowler)
  - Blueprint for a system, the implicit high level plan for its construction
  - "set of design decisions which, if made incorrectly, may cause your project to be cancelled." (Eoin Woods)

HERMES
A HUMAN WAY OF MANAGING SERVICES

What is Architecture ?

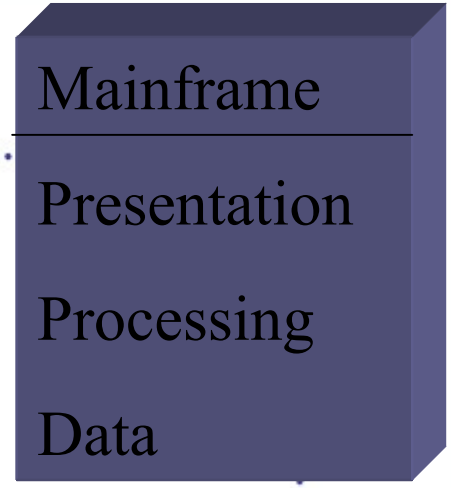
- The software architecture of a program is the structure of the system, which comprises
  - software elements
  - the externally visible properties of those elements
  - and the relationships among them.

What is SOA ?

HERMES
A HUMAN WAY OF MANAGING SERVICES

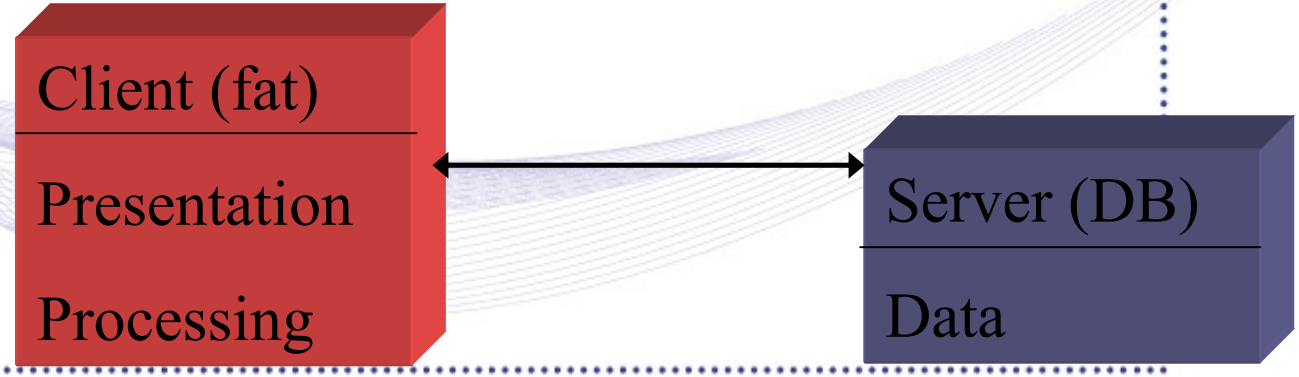
Architectures : historics

- Central systems : mainframes
  dumb terminals ...



- Client-server (2-tier) (screenscraping)

PB, Natstar,
VB, TFM,
Delphi

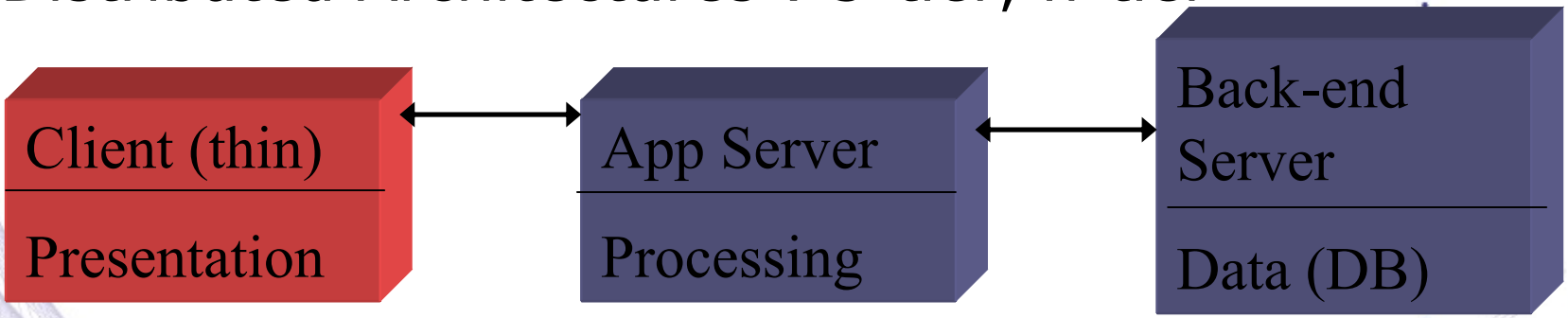


Architectures : historics

- 2-tier client-server : many problems
  - PCs must be regularly replaced
  - Maintaining PCs
  - Software distribution + updates
  - Hell if installation at the clients : support, multitude of configs to support, soft distribution ...
  - Security : everybody has access to the DB, ex : homebank software
  - Network traffic
  - Integration after merges
  - …

Architectures : historics

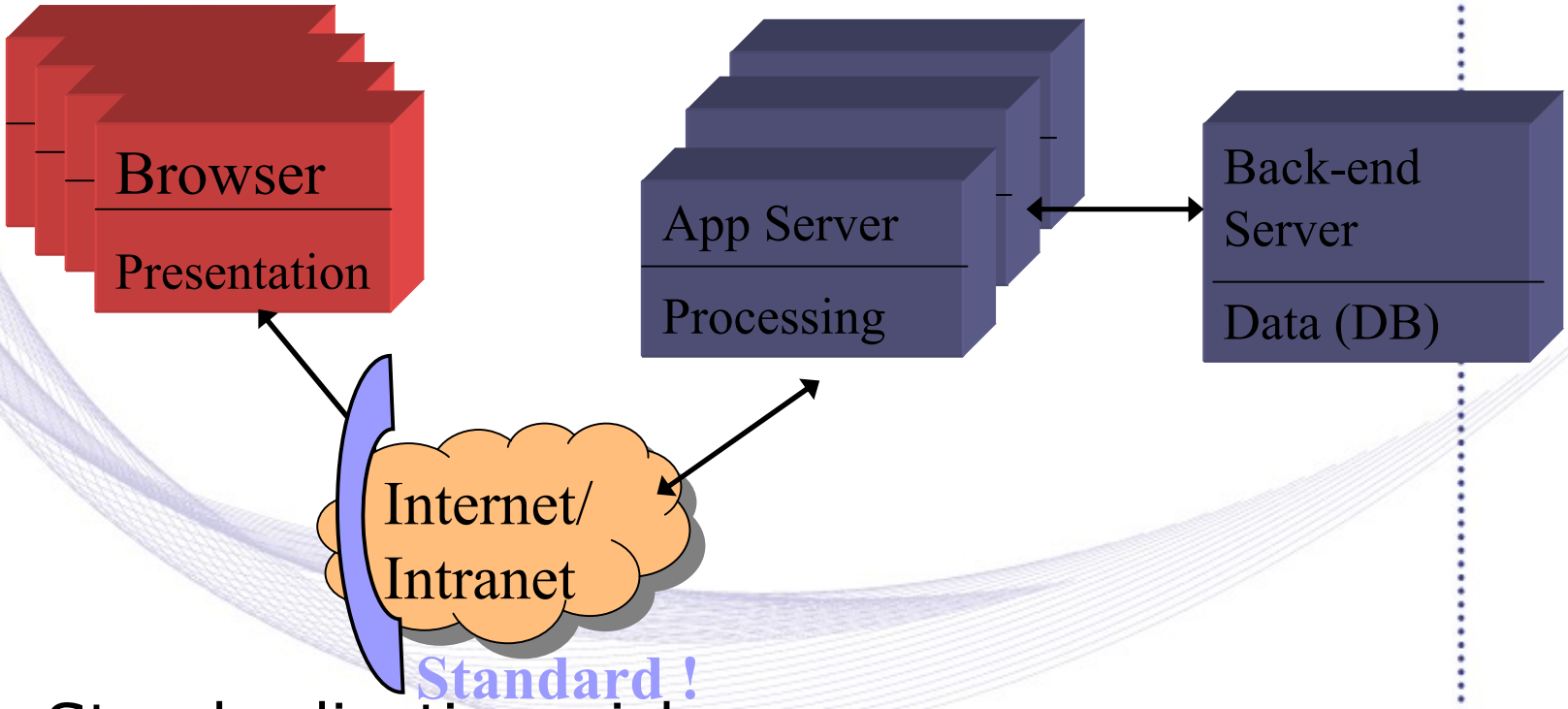
- Distributed Architectures : 3-tier, n-tier



- Components approach : n-tier
- Thin client : well adapted for internet, e-commerce... But not only

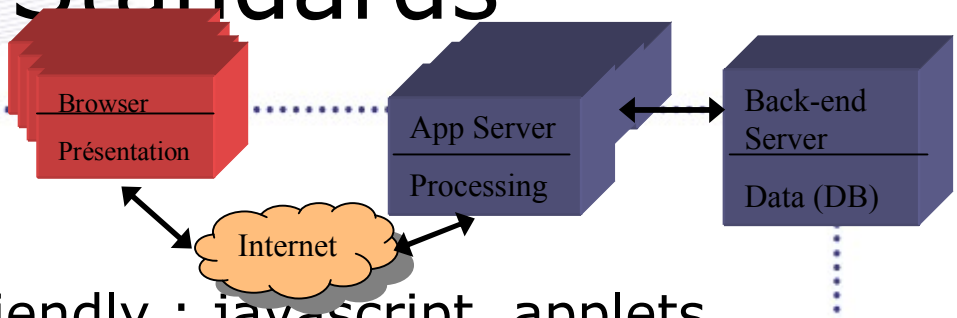
Architectures : historics

- Typical internet architecture



- Standardisation wish

Inter/Intra-net Standards



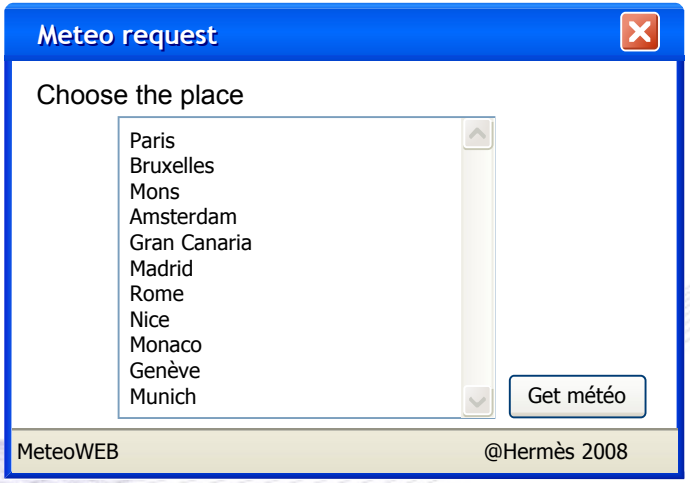
- Client
  - HTML browser
  - Add-ons for + user-friendly : javascript, applets (+/- std) + to reduce network traffic (local ctrls before sending to the server)
- Internet : Http on TCP/IP protocol
- DB server : Relational DB, SQL +/- standard
- Application servers : JSP, Servlets, EJB…
- All this defines J2EE

What is SOA ?

- SOA is
  - Software architecture,
  - Based on 4 key elements
    - Application front-end (owner of the business process)
    - **Service** (business functionality, structure of SOA)
    - Service repository (catalog of services)
    - Service bus (interconnection)
  - Business services (not technical) :
    - No impact from technology on the high-level structure
    - Technology cannot cause dependencies between components

Example

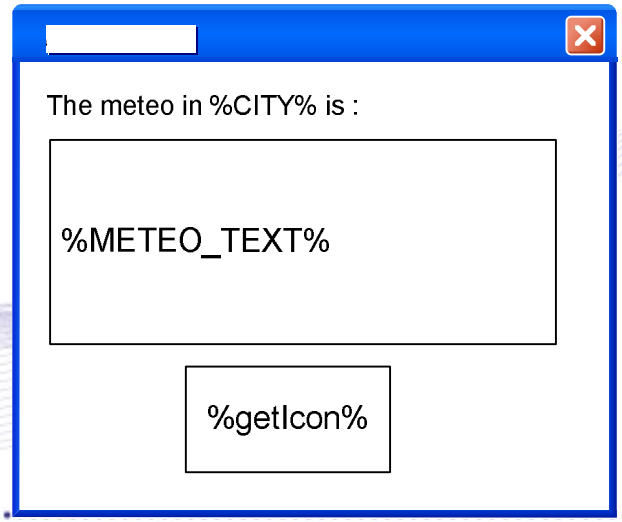
- An application which gives you the meteo.
- A first page asks you a place



- On click : http request **getMeteo(city)**

Example

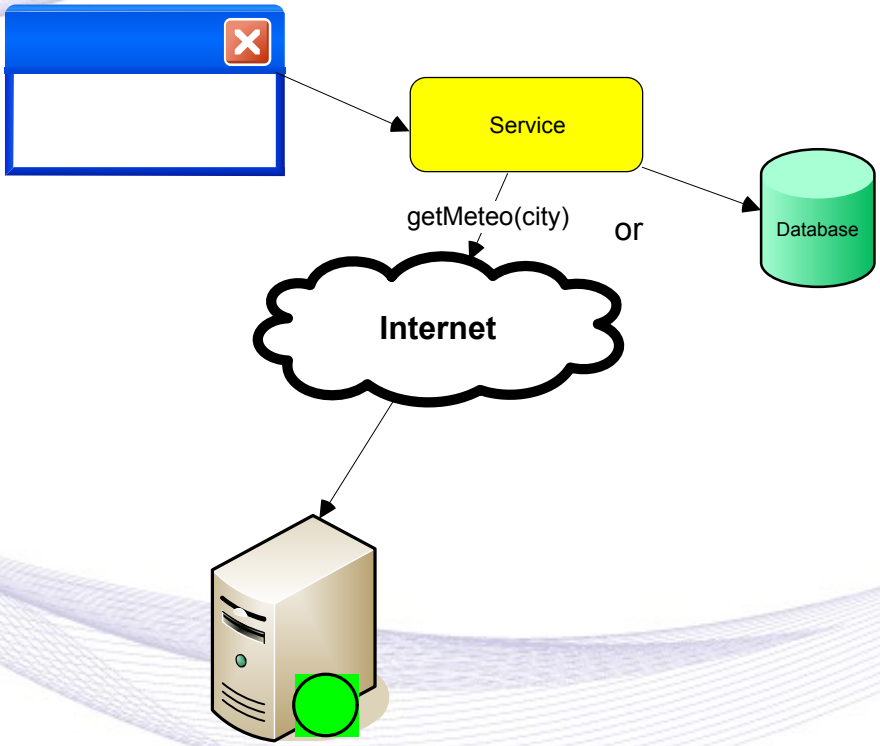
- Server
  - receives the request
  - determines which page must be built
  - calls a new (html) page and…
  - fills it



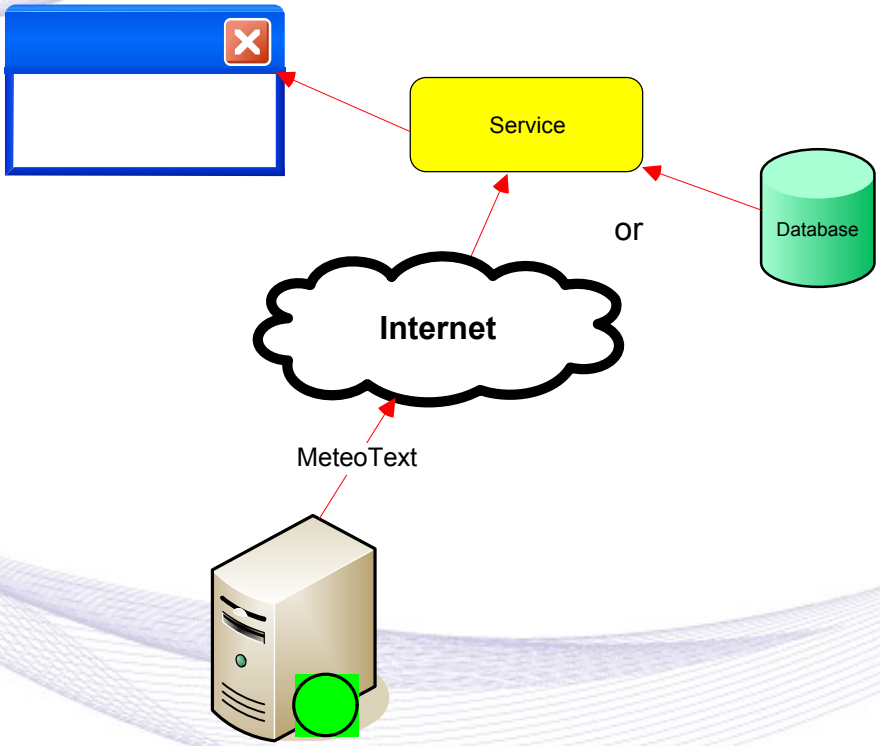
Example

- Meteo_text can come from whatever other specialized site
- Mix GUI and business code : BAD !
- Create a “service” to obtain the information

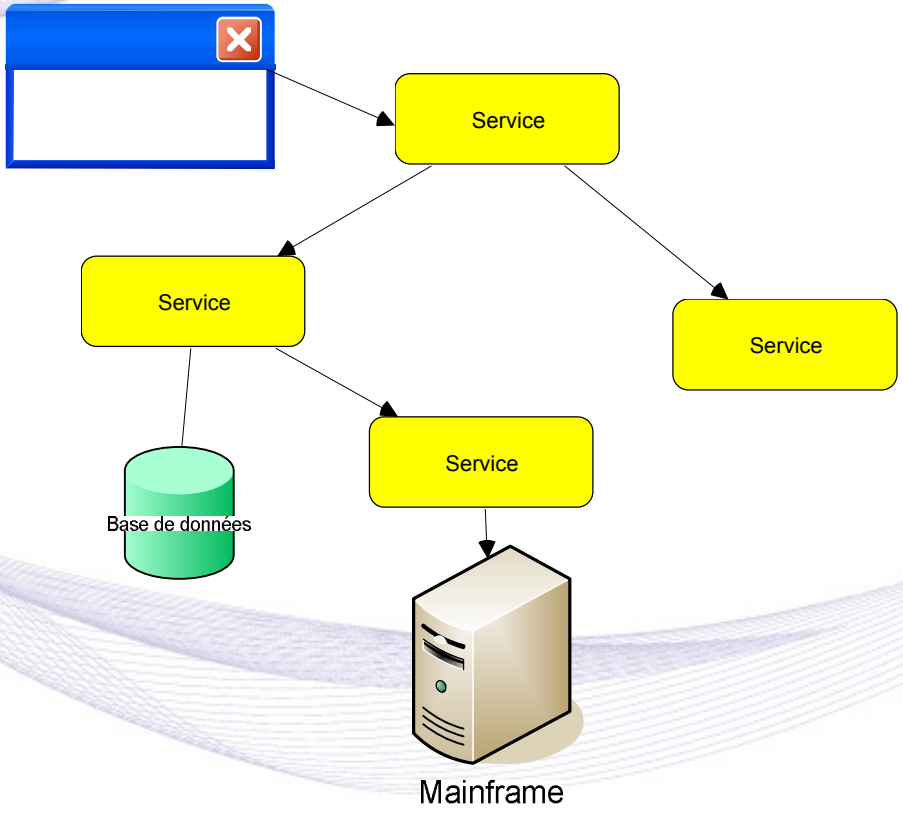
Example



Example



Generalizing example

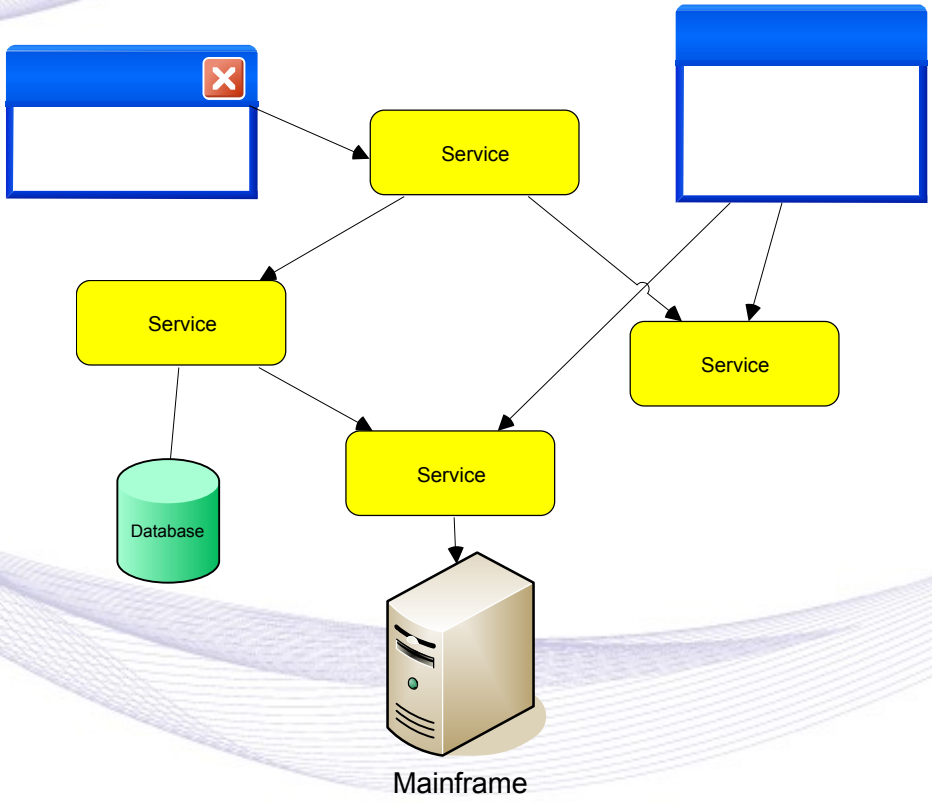


HERMES
A HUMAN WAY OF MANAGING SERVICES

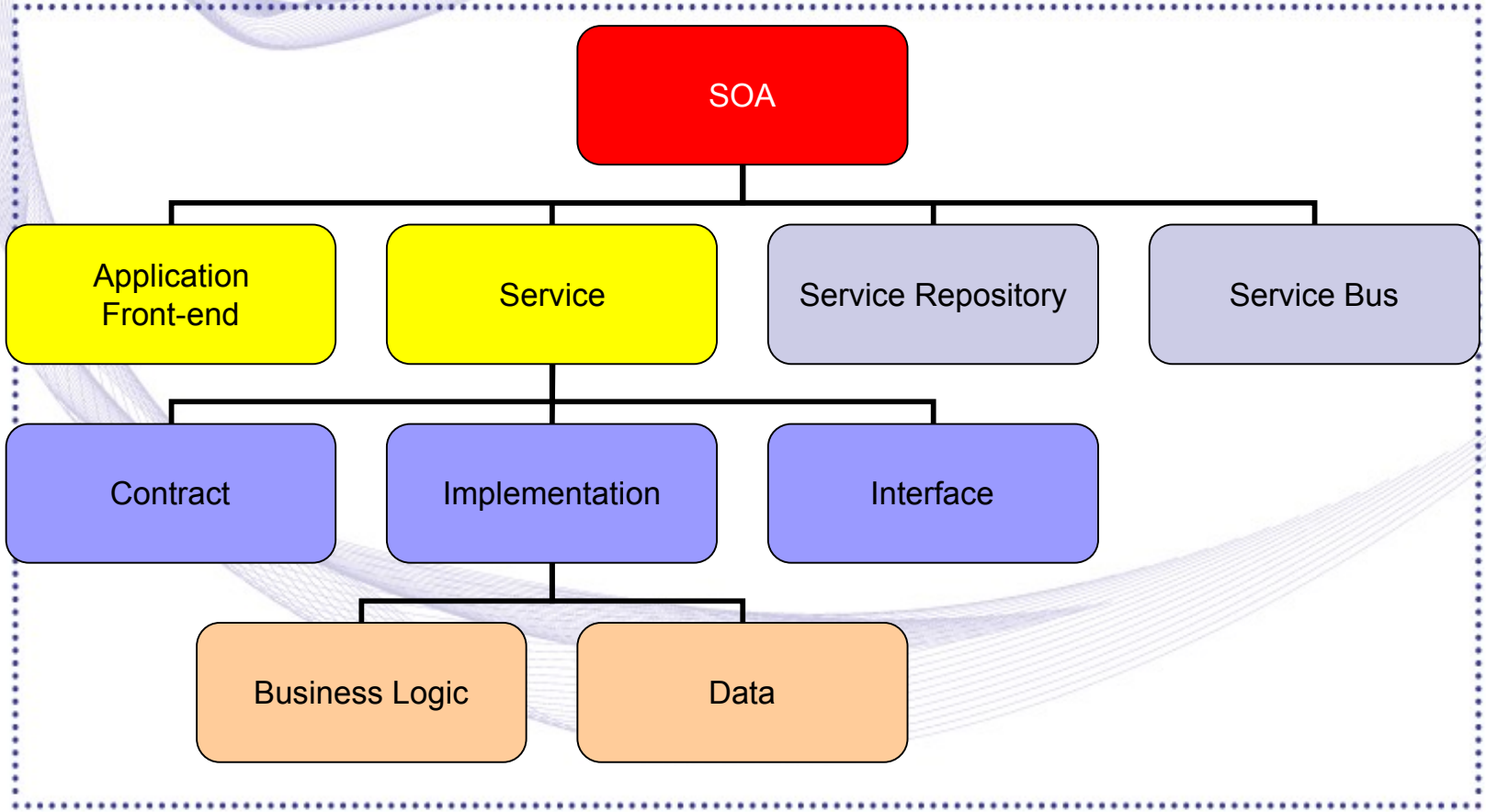
Why is it bad to mix GUI and business ?

- Gui can change due to :
  - Process reengineering
  - Restructuration, acquisition…
  - Relooking, revamping
- Business process will change over time
- But the business services do not change !
- Ex : approval of mortgage credits in banks

Other advantage : reusable services

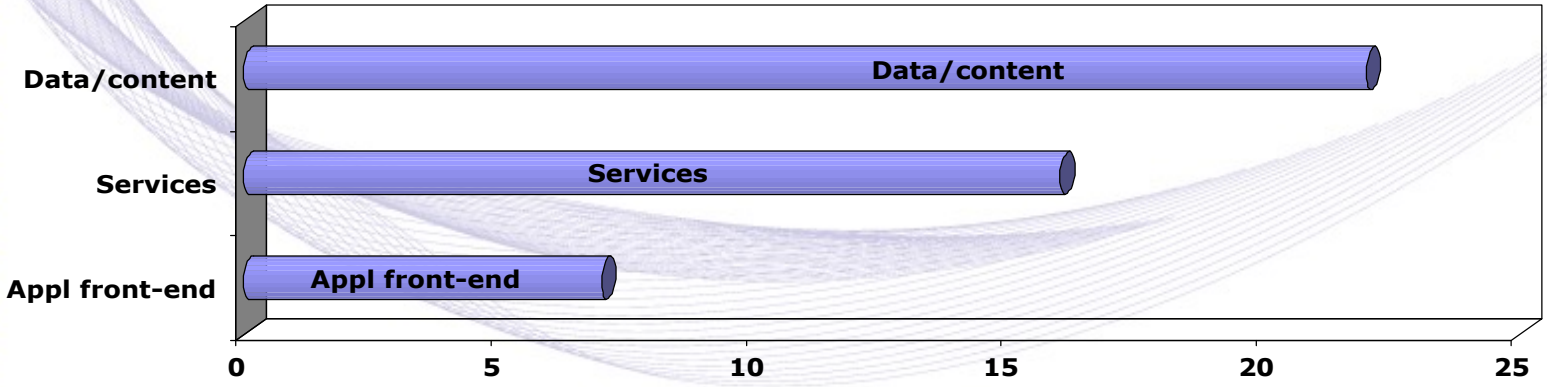


SOA concepts



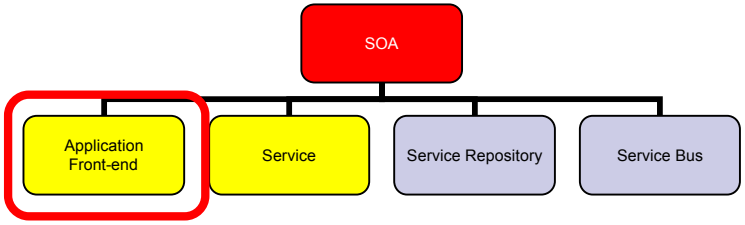
Services

- Software component of functional meaning
- Encapsulates a high-level business concept
- Much more stable than processes or applications
- does not depend on the context or state of other services (stateless, as much as possible)

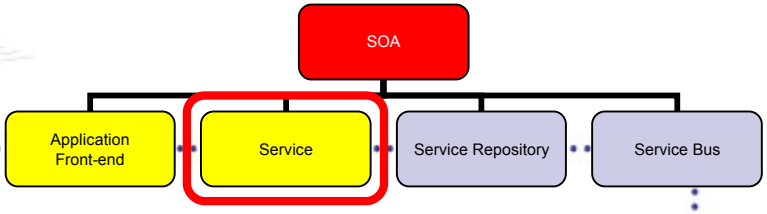


Application Front-ends

- Web applications
- Rich clients
- But also batch
- Delegate the functionality to Services
- Initiate the business process
- Receive the results



Services



Before

System

Data | Functionality

Now

System

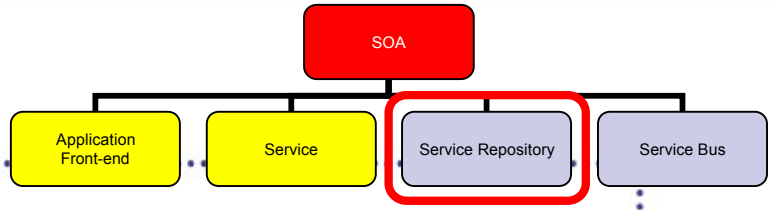
Application Front-end | Service

Data | Functionality | Data | Functionality

OR

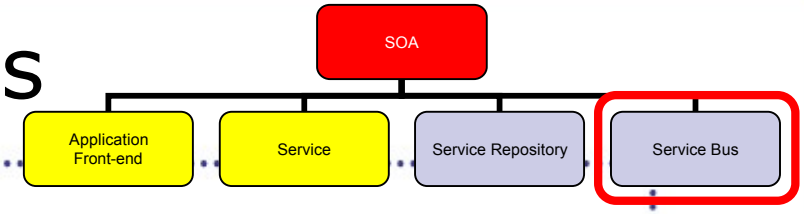
Object | Component | Object | Component

Service Repository



- Facility to discover services and acquire information to use them
- Contains the content of the contract
- May also contain informations as
  - Physical location
  - Contact persons
  - SLA
  - Security issue
  - Special constraints…
- Necessary for long term enterprise SOA
- Organization : under responsibility of an architecture board

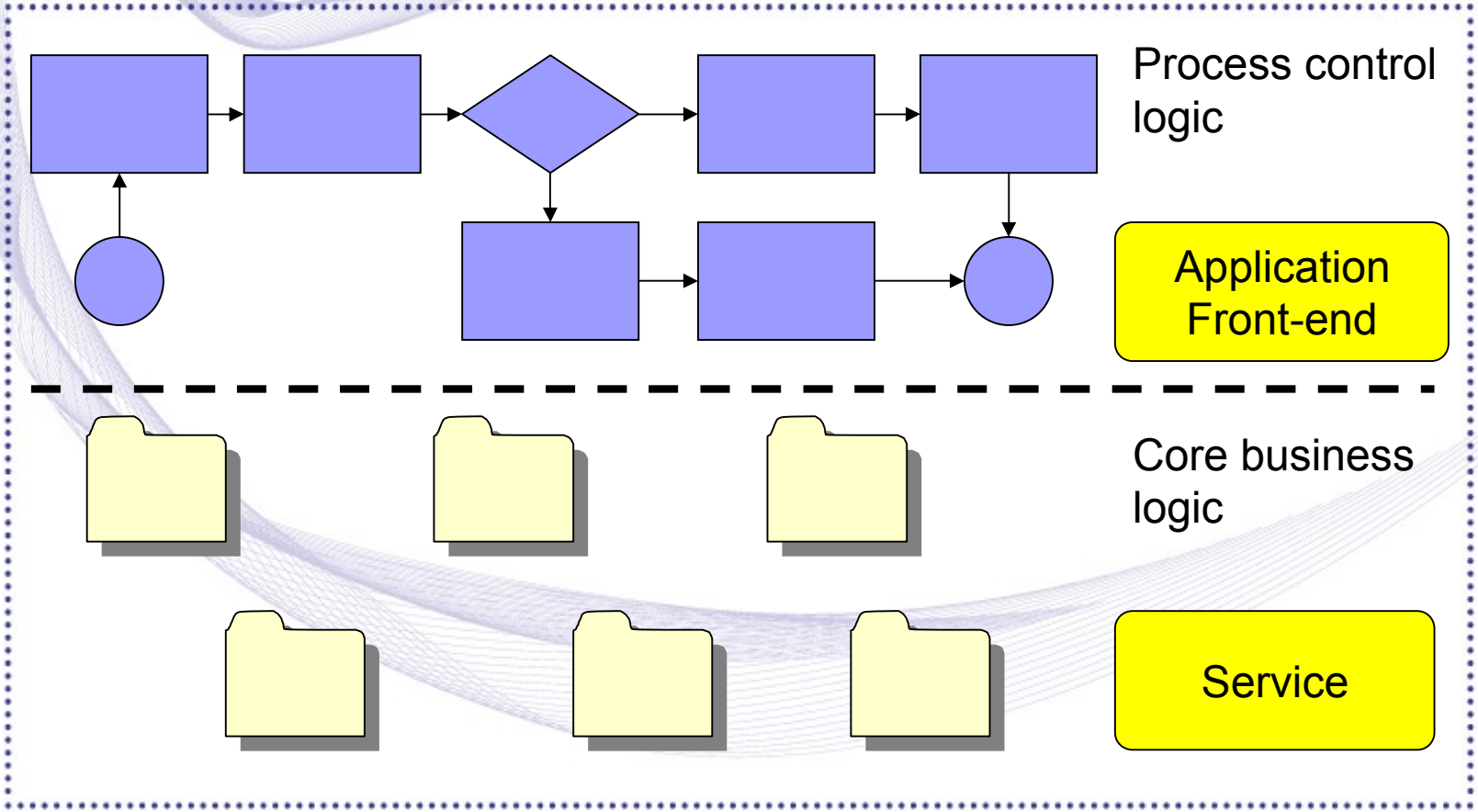
Enterprise Service Bus



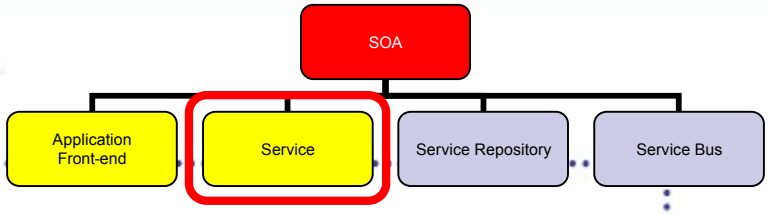
- Connects
  - Services
  - Application front-ends
- Provides
  - Connectivity (intelligent routing)
  - Heterogeneity of technology (e.g. data transformation)
  - Heterogeneity of communication concepts (more asynchronous if possible, uncoupling is the master word)
  - Technical services : logging, auditing, security, transactions…

HERMES
A HUMAN WAY OF MANAGING SERVICES

Enterprise Service Oriented Architecture



Particular case : WebServices

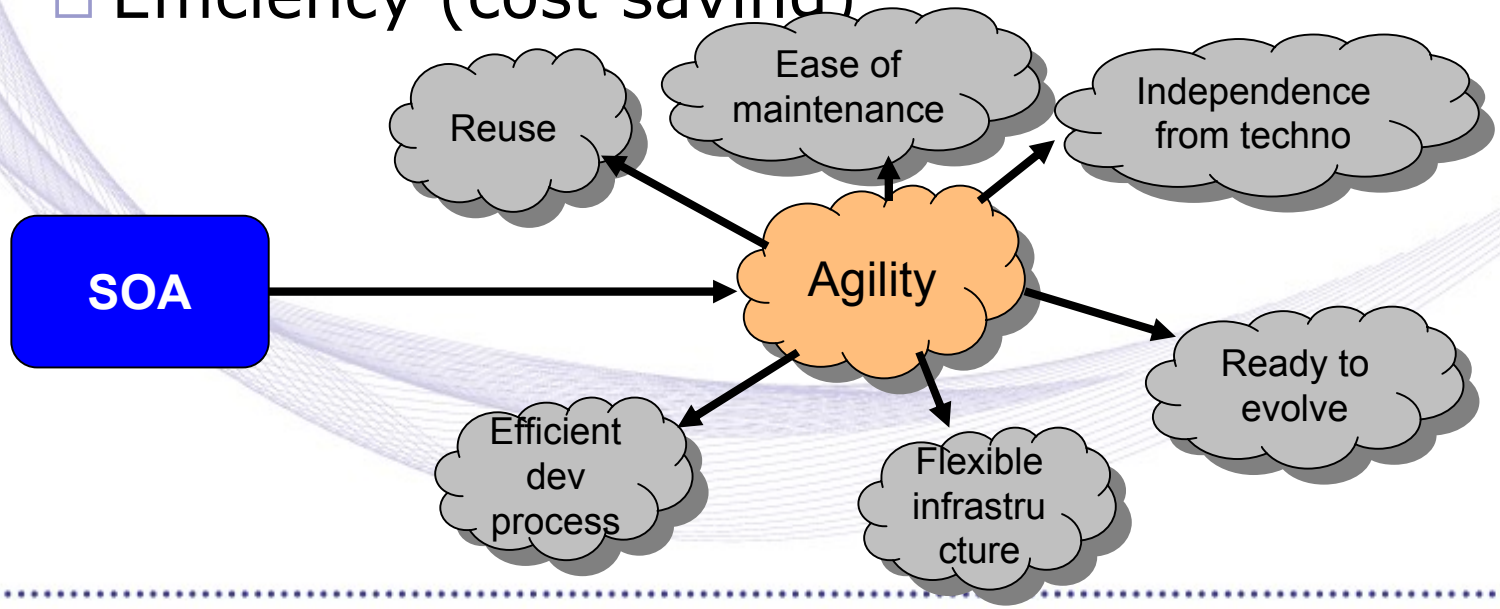


- Definition :
  - Business components
  - Self sufficient
  - Self descriptive
  - Called via internet/intranet
  - Respecting quality agreements
  - Based on XML

Why SOA ?

SOA Benefits

- SOA main drivers
  - Agility
  - Efficiency (cost saving)



SOA benefits for people

- CEO
  - Better reaction for business demands
  - Shorter term planning (step by step app.)
  - Budget : less maintenance
  - Techno independence
- CIO
  - Techno independence
  - IT becomes “enabler” to the business
  - Manageable project size
  - Manage heterogeneity

SOA benefits for people

- Architect
  - Very interesting tasks
  - Opens real opportunities to build
  - Loose coupling
  - Code reuse
- Project Manager
  - Smaller & shorter projects
  - Parallel development
  - Reduced risk
  - Easier testing & integration

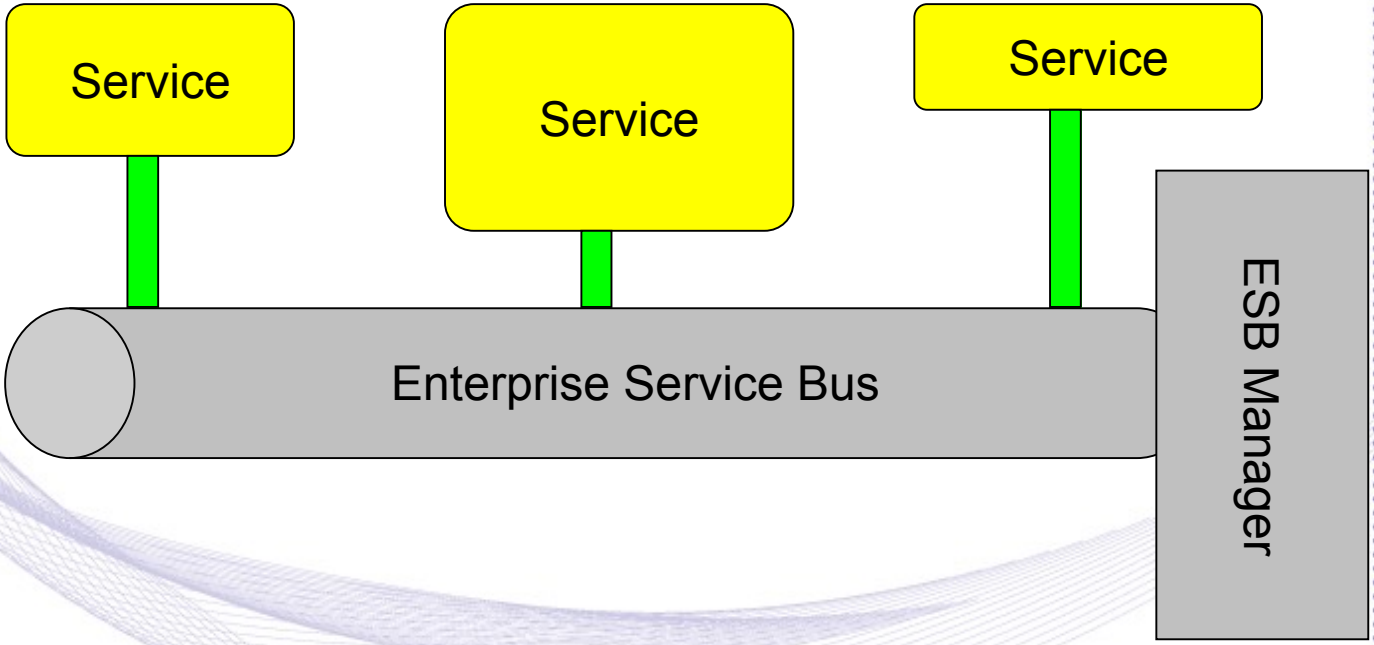
SOA benefits for people

- Software developer
  - Reduction in dependency
  - Rapid prototyping
  - Better defined requirements
  - Simplified testing (loose coupling)

Implement the architecture

HERMES
A HUMAN WAY OF MANAGING SERVICES

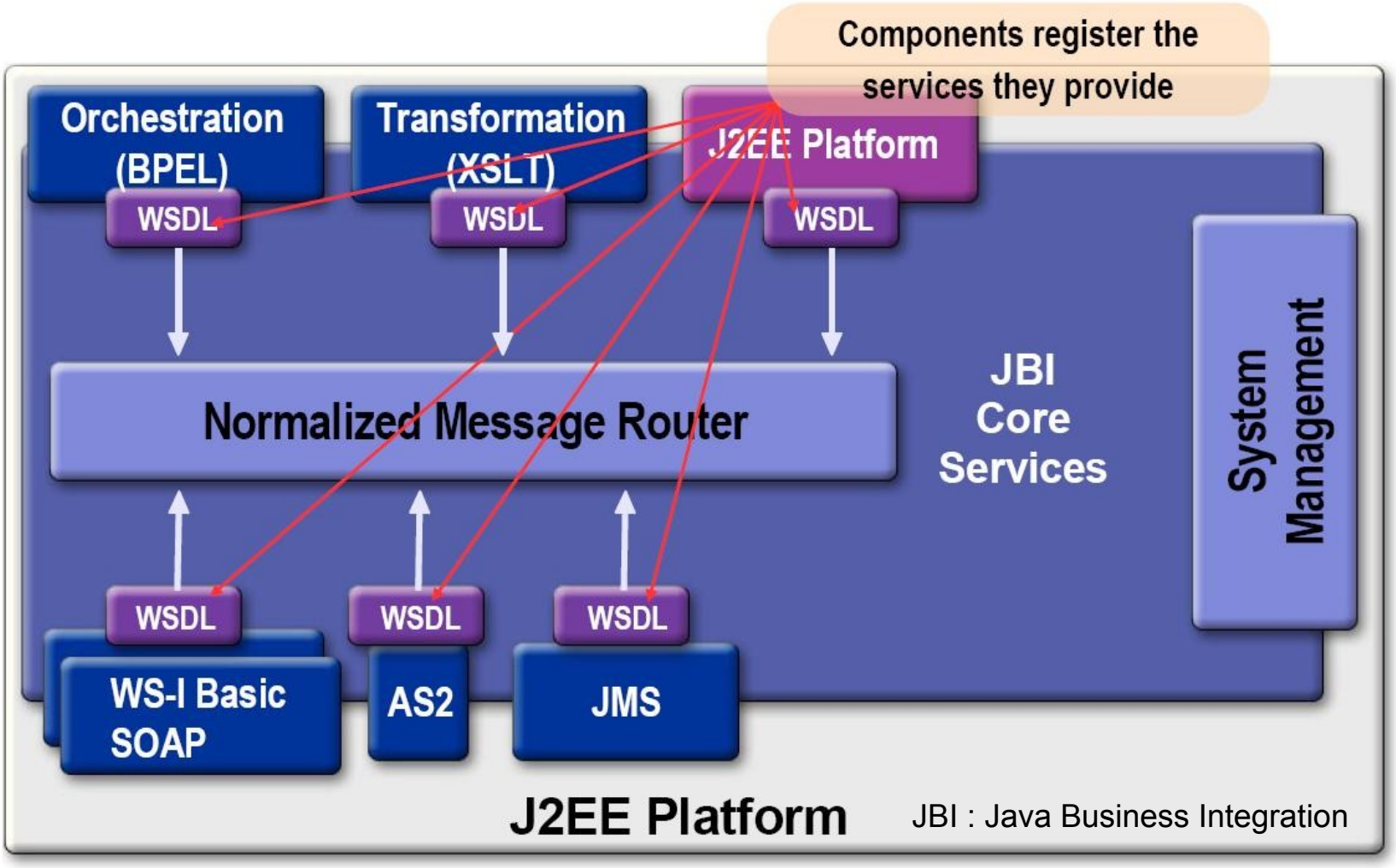
SOA Architecture : ESB



HERMES
A HUMAN WAY OF MANAGING SERVICES

What is ESB ?

- Message broker
- Enables the implementation of SOA
- Tries to remove the coupling between the service called and the transport medium
- Brings transportation and routing
- Provides an abstraction for endpoints



JBI : Java Business Integration

HERMES
A HUMAN WAY OF MANAGING SERVICES

Benefits of an ESB

- Allows Pluggable Component (with data transformation)
- JBI mediator : isolates the services that don't have to know each other = **Normalized Message Router**
- Un-coupling applications to services and service to service
- Suppresses the multiple point to point connections (not scalable & hard to maintain; impact analysis easier)
- OpenSolution : vendor independent, no lock-in for small providers

HERMES
A HUMAN WAY OF MANAGING SERVICES

Capabilities of an ESB

Invocation	Synchronous & asynchronous, service locating
Routing	Addressing, static/content- or rules- or policy-based routing
Mediation	Adapters, protocol transformation
Messaging	Message processing, transformation
Aggregator	Multiple implementations of a service, exposed as a single one
QoS	Security (encrypt, sign), guaranteed delivery, transaction management
Management	Monitoring, auditing, logging, …

HERMES
A HUMAN WAY OF MANAGING SERVICES

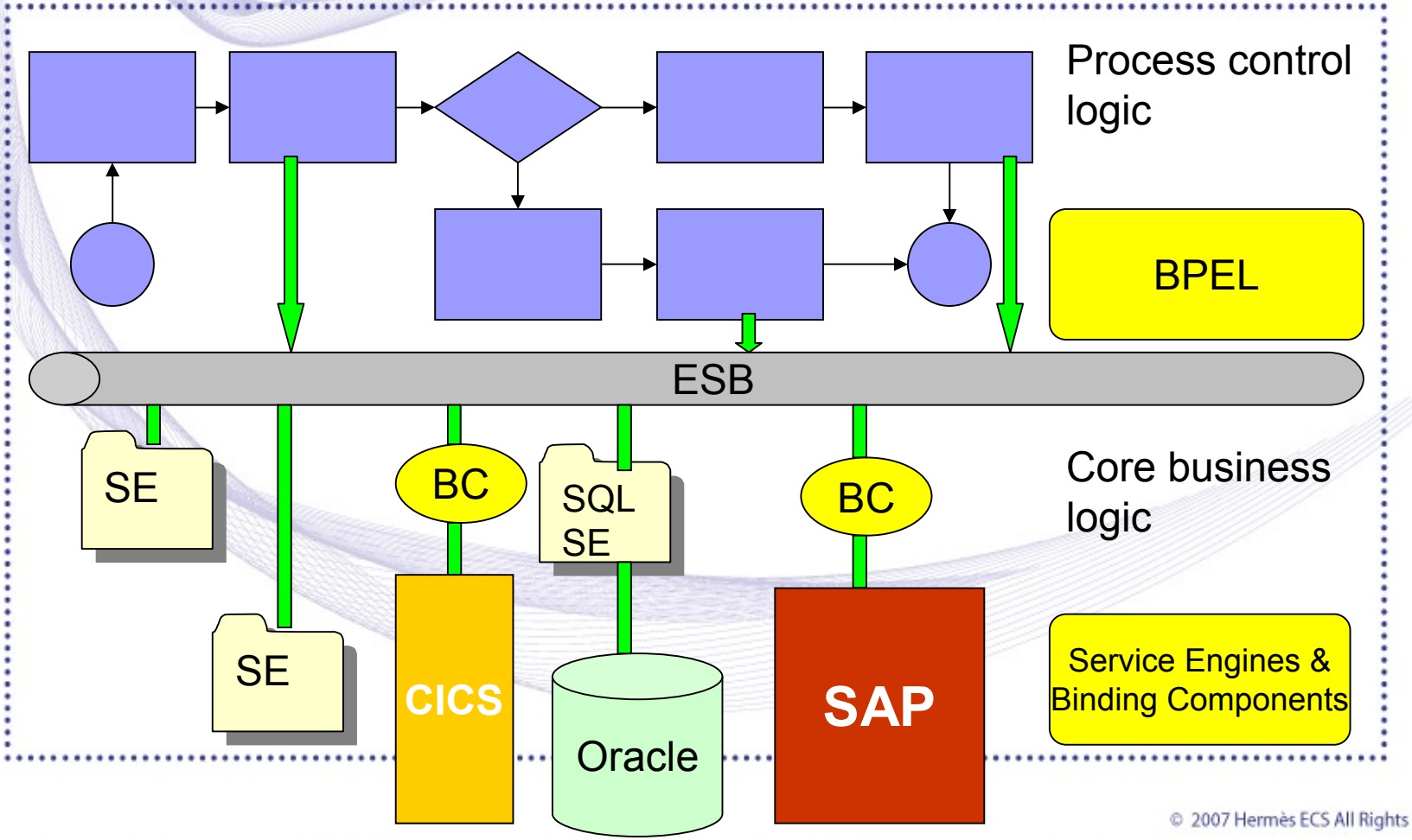
Common characteristics of ESB

- OS & language agnostic
- Often uses XML
- Should support WebServices
- Multiple messaging patterns : synchronous & asynchronous request/reply, fire & forget, publish/subscribe…
- Transformation services : e.g. XSLT
- Message validation against “schema”
- Supports queuing, holding messages if apps unavailable

ESB = glue between services

- BPEL : Business Process Execution Language : defines the workflow of services (orchestrator)
- Services : provide the business rules and the business basic behavior, are (possibly) reusable
- BPEL + services = an application for the user

HERMES
A HUMAN WAY OF MANAGING SERVICES



HERMES
A HUMAN WAY OF MANAGING SERVICES

ESB conclusions

- New technology based on **experience from the past** (Middleware, EAI, stable architectures…)
- Open products exist & are well adopted
- JBI & ESB receive **strong support from SUN**
- Based on **solid architectural principles**
- Perfect addition for a sound SOA implementation
- Facilitates the integration & future evolution by providing a very **open architecture**

HERMES
A HUMAN WAY OF MANAGING SERVICES

SOA Analysis

HERMES
A HUMAN WAY OF MANAGING SERVICES

How to analyze for SOA ?

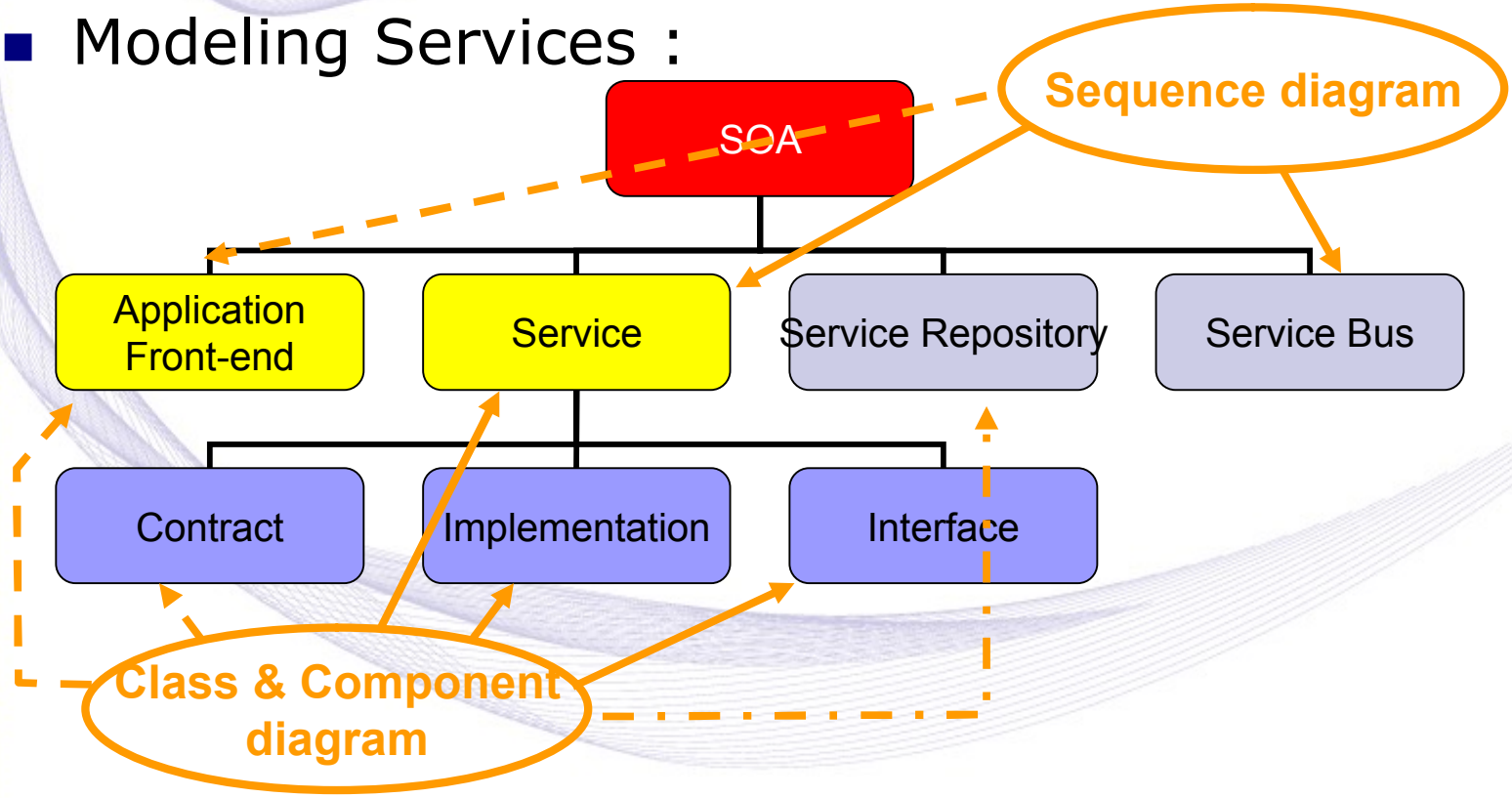
SOA Analysis

- Modeling is really important
- Analysis world : ONE standard
  - Basis for services repartition
  - Model inter-dependencies
  - Know what to develop first
  - Ensure services are reusable
  - Build the SO Architecture !

UNIFIED MODELING LANGUAGE

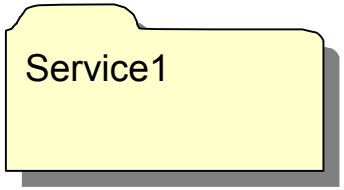
SOA and UML

- Modeling Services :

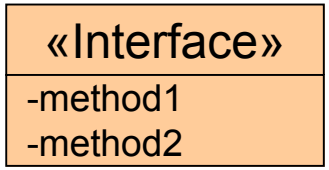


UML class & component diagram

- Services <-> UML packages

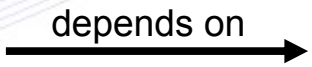


- Interfaces <-> Interface classes

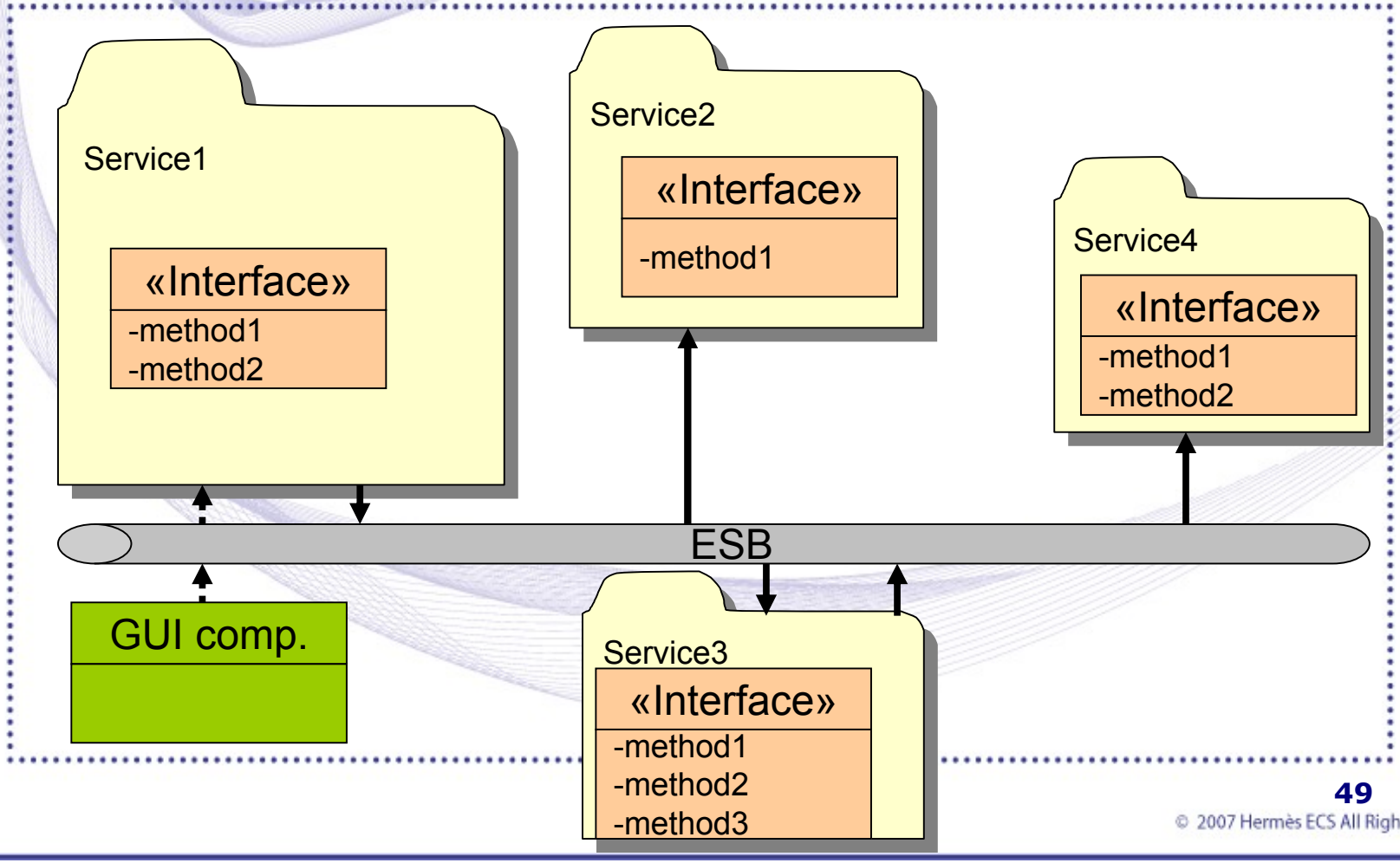


- Contracts : documentation

- Dependency betw. Services <-> UML dependency



Example

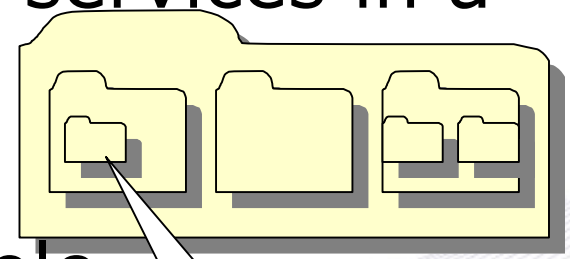


UML class & component diagram

- May also be used for the

Service Repository

- You can organize your services in a hierarchy

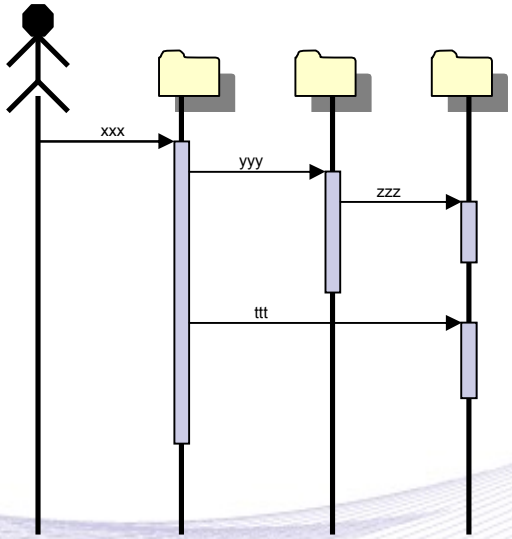


- In a CASEtool, the whole documentation of the services is just one-click away

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor. Cras elementum ultrices diam. Maecenas ligula massa, varius a, semper congue, euismod non, mi.

UML Sequence Diagram

- Model the dynamic part, the **interactions**



- **What will happen on the bus**

UML conclusions

- **Class & component diagrams** (static)
  - View of what services are available
  - Documentation of the interface
  - Shows the dependencies between services
- **Sequence diagrams** (dynamic)
  - View on how the services communicate
  - View on how the appl. uses the services
- UML analysis is very well suited to SOA

Conclusions

Conclusions

- SOA can bring BENEFITS
- SOA is an evolution, not a **R**evolution
- SOA does not require technology change !
- SOA adapts to all technologies (including legacy)
- You can transition smoothly to SOA

- … but SOA needs a clear decision & management support !

Conclusions

- To implement SOA, you need good modeling :
  - Analysis
  - Architectural process
- UML is the ideal candidate to support this modeling
- SOA is **maturity** after C/S, n-tier, CBD…

HERMES
A HUMAN WAY OF MANAGING SERVICES

QUESTIONS ?

E-mail: frederic.vermaut@hermes-ecs.com
Web site: http://www.hermes-ecs.com