Web services

Service oriented architecture (SOA)

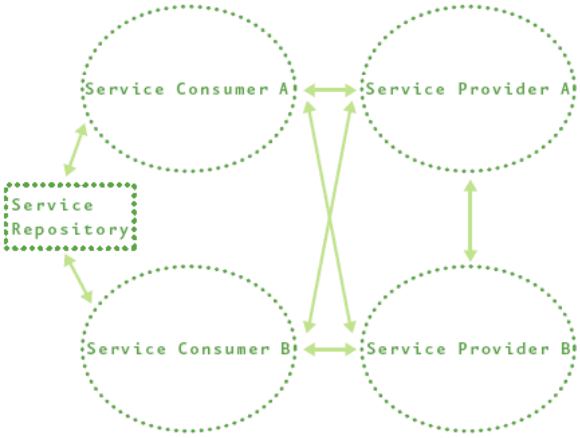
Th.S Cao Thái Phương Thanh
caothaiphuongthanh@gmail.com

Last content

- SOA introduction:
  - SOA is an IT architecture strategy for business solution (and infrastructure solution) delivery based on the concept of service-orientation.
  - It is a set of components which can be invoked, and whose interface descriptions can be published and discovered.
  - It aims at building systems that are extendible, flexible and fit with legacy systems.
  - It promotes the re-use of basic components called services.

Software engineering for SOA

- SOA components:
  - Service providers
  - Service consumers
  - Service repository

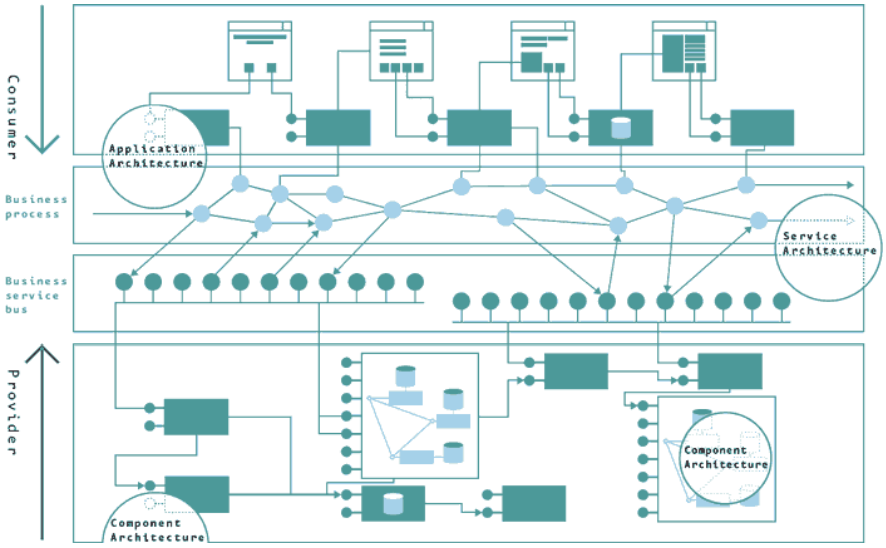


Software engineering for SOA

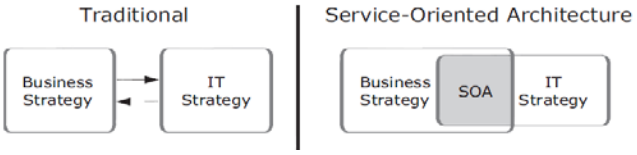
- SOA architectures:
  - The Application Architecture: this is the business facing solution which consumes services from one or more providers and integrates them into the business processes.
  - The Service Architecture: this provides a bridge between the implementations and the consuming applications, creating a logical view of sets of services which are available for use, invoked by a common interface and management architecture.
  - The Component Architecture: this describes the various environments supporting the implemented applications, the business objects and their implementations.

Software engineering for SOA

- SOA architectures:



Software engineering for SOA



- SOA core reason: business driven
  - The business drives the services, and the services drive the technology.
  - In essence, services act as a layer of abstraction between the business and the technology.
  - SOA architect must understand the dynamic relationships between:
    - Business and Service
    - Service and Technology

Software engineering for SOA

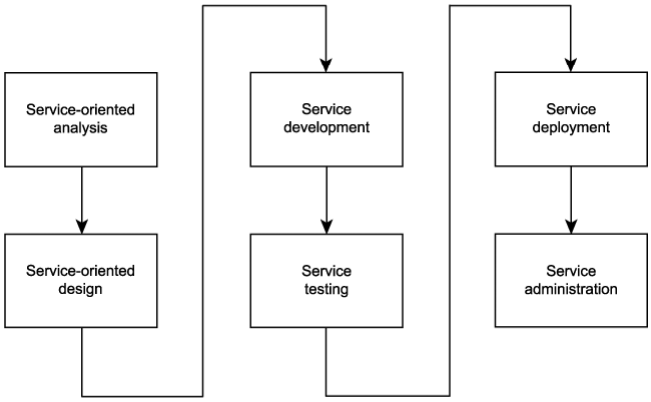
- SOA principles:
  - **formal contract**: For services to interact, they need not share anything but a formal contract (i.e. interface) that describes each service and defines the terms of information exchange.
  - **loosely coupled**: Services must be designed to interact without the need for tight, cross-service dependencies.

Software engineering for SOA

- SOA principles:
  - **abstract underlying logic**: The only part of a service that is visible to the outside world is what is exposed via the service contract. Underlying logic, beyond what is expressed in the descriptions that comprise the contract, is invisible and irrelevant to service requestors.
  - **autonomous**: The logic governed by a service resides within an explicit boundary. The service has control within this boundary and is not dependent on other services for it to execute its governance.

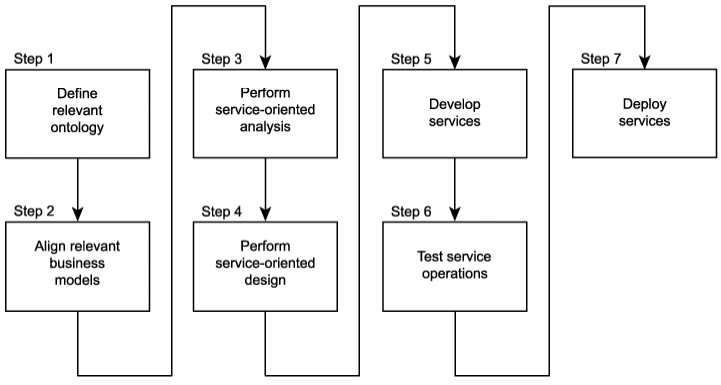
Software engineering for SOA

- SOA lifecycle:



Software engineering for SOA

- SOA top – down approach:



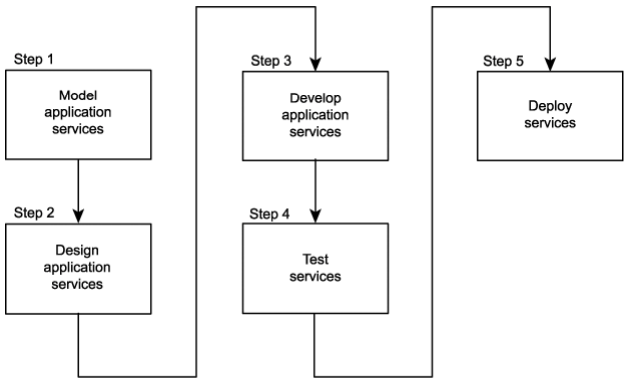
- analysis first to create overall business and application services
- might lead to high quality SOA but also impose lots of analysis work

Software engineering for SOA

- SOA top – down approach example:
  - An company used the top – down approach when they first ventured into building services for their B2B solution. They went through the process of defining a corporate ontology and then implementing the results within their business models and their service designs. The result was a set of business and application services, highly standardized and very reusable that quickly accommodate changing business requirements.
  - The time it took to carry through the top-down analysis, though, was significant. Despite the success of the top-down strategy, IT managers are hesitant to undergo this process again.

Software engineering for SOA

- SOA bottom – up approach



- services are built on as needed and modeled to encapsulate application logic to serve the immediate requirements of the solution.
- easy to follow but might not take advantage of service – oriented

Software engineering for SOA

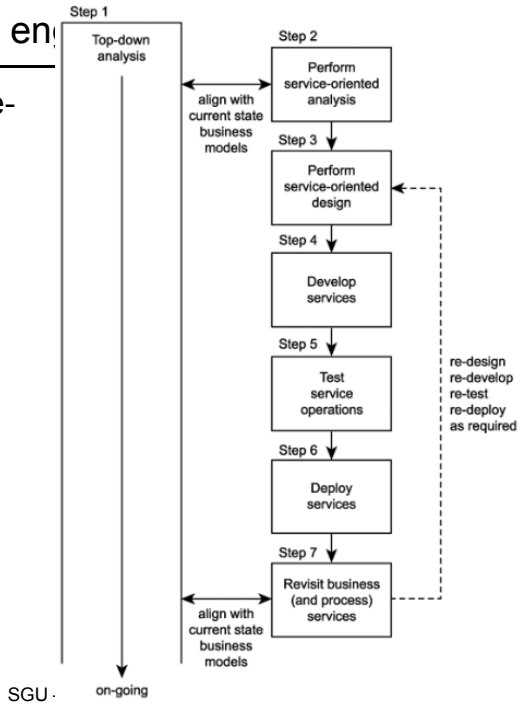
- SOA bottom – up approach example:
  - To fulfill partners's ebusiness requirements, a company built web services which wrap its legacy systems. These services accommodate specific needs and therefore fulfilled business requirements in short time.
  - However, if the company changes its own applications (e.g. upgrade a module within the existing accounting system); or the partners introduce new specifications; these bottom – up web services might require significant modifications.

Software engineering for SOA

- SOA meet-in-the-middle approach:
  - Begin the top-down analysis, focusing first on key parts of the ontology and related business entities
  - When the top-down analysis has sufficiently progressed, perform service-oriented develop while the top-down analysis is going on
  - As the top-down analysis continues to progress, revisit business services

Software en

- SOA meet-in-the-middle approach:



SOA model

1. Service identification
2. Service specification
3. Service composition
4. Service implementation

SOA model

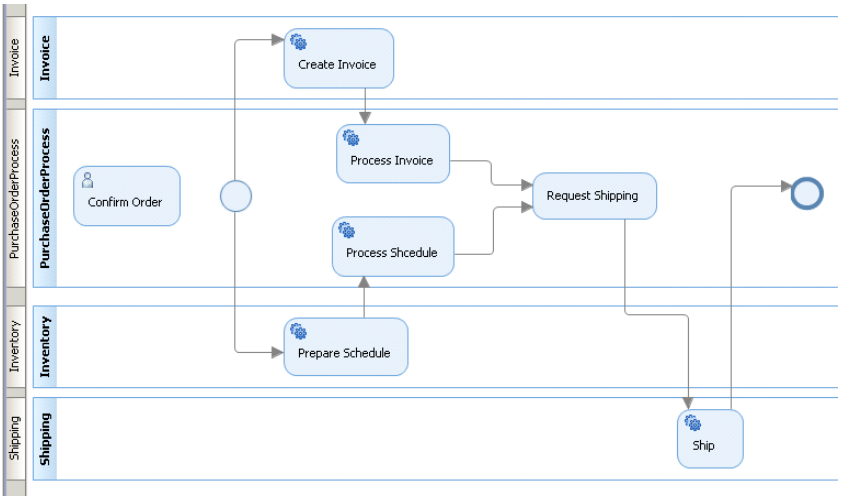
- We use Ms Visio, IBM Rational Software Architect to model services (service specifications)
- Then use Microsoft Windows Communication Foundation (WCF) to implement the model (i.e. to create SOA system)

SOA model

- Model SOA example: Purchase Order Process
  - After customers confirmed the purchase order, processing starts by creating a invoice from the order.
  - At the same time, the order is sent to inventory to prepare the products will be shipped (e.g. when the products will be available and where they are).
  - After both the invoice and production schedule are ready, the process requests shipping service.

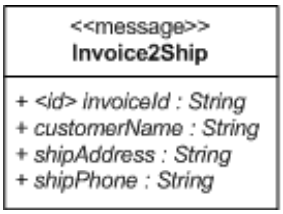
SOA model

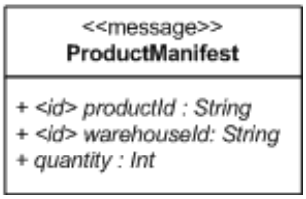
- Model SOA example: Purchase Order Process



SOA model

- Model SOA : Shipping Service Specification

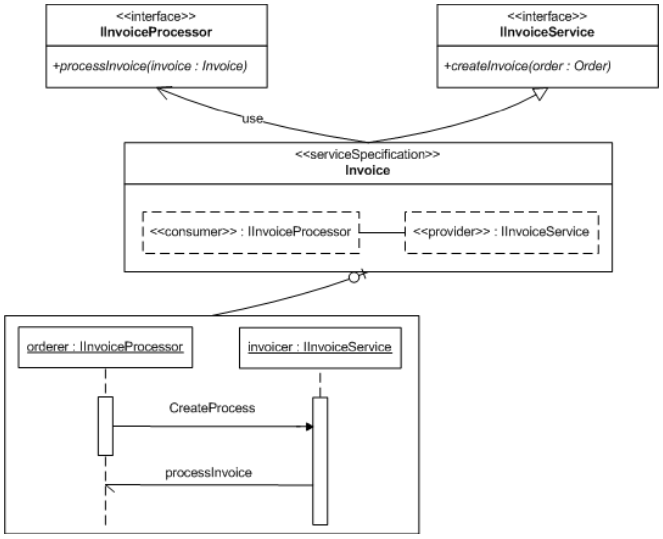




- Note: message types are serialized and exchanged between services - consumers

SOA model

- Model SOA : Invoice Service Specification

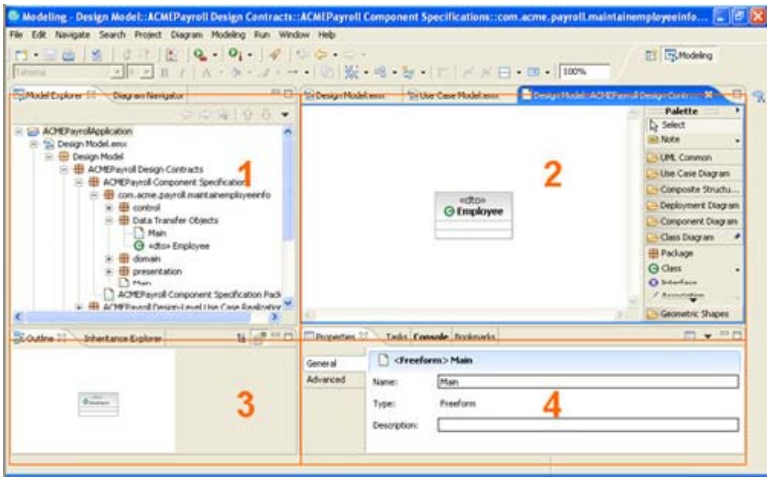


SOA model

- Beside service specifications, other components are modeled as usual (i.e. standard UML)
- IBM provides UML 2.0 profile for Software Services which is integrated in IBM Rational Software Architect. However, the tool is not available freely.

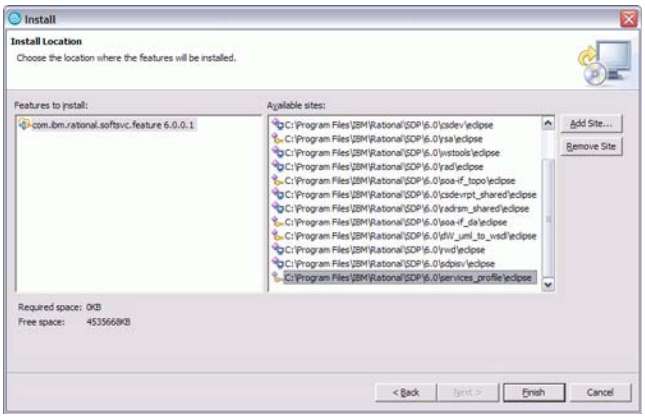
SOA model

- IBM Rational Software Architect tool with UML 2.0 profile for Software Services



SOA model

- IBM Rational Software Architect tool with UML 2.0 profile for Software Services
  - Install the service profile



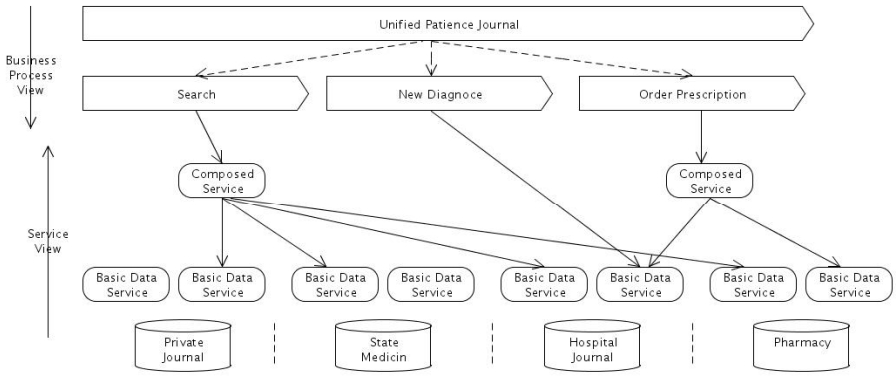
Pratice

- Very technically, SOA is built of reusable block:
  - Process Services – complex business logic
  - Composed Services - business logic.
  - Basic Data Service - CRUD operations.

[Ref 4]

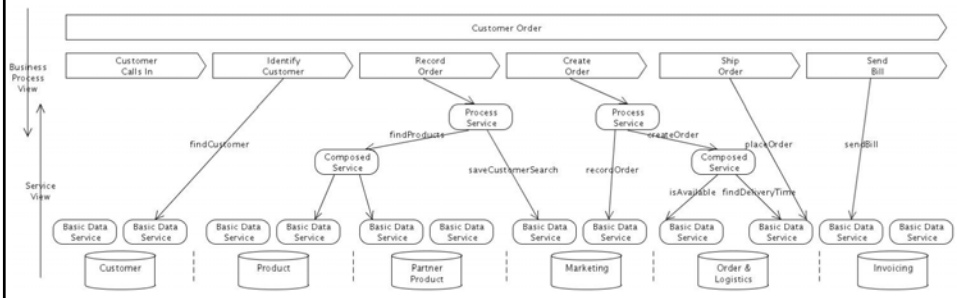
Pratice

- Process Services – complex business logic
- Composed Services - business logic.
- Basic Data Service - CRUD operations.



Pratice

- Process Services – complex business logic
- Composed Services - business logic.
- Basic Data Service - CRUD operations.



Pratice

How to design the conversation

Situation 1: Wrong way

Requester: "Get me Amy's check account balance"
Provider" "$100"
Requestor: "What is her credit limit?"
Provider: $1000000

What is wrong? Provider must remember Amy's account between 2 requests

Situation 2: Better way

Requester: "Get me Amy's check account balance"
Provider" "$100"
Requestor: "What is Amy's credit limit?"
Provider: $1000000

Provider does not have to remember Amy's account between 2 requests

Next

- Implement SOA model using WCF

References

1. Thomas Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005
2. Simon Johnston, UML 2.0 Profile for Software Services, 2005
3. Magnus K Karlsson, SOA by Examples, 2009, Personal blog
4. Johan den Haan, The Process Centric vs. Information Centric approach to SOA, 2010, Personal blog

Self - learning

- IBM developer Works
  - http://www.ibm.com/developerworks/views/rational/libraryview.jsp?search_by=Design+SOA+services+with+Rational+Software+Architect,+Part
  - http://www.ibm.com/developerworks/rational/tutorials/dw-r-modsoacase5/index.html
  - http://www.ibm.com/developerworks/views/rational/libraryview.jsp?search_by=Modeling+SOA: