IT4409: Web Technologies and e-Services
2018-2

Service-Oriented Architecture (SOA)

Instructor: Dr. Thanh-Chung Dao
Slides by Dr. Binh Minh Nguyen

Department of Information Systems
School of Information and Communication Technology
Hanoi University of Science and Technology

TOPICS

Introduction to SOA
- Service
- Service Oriented Architecture

Web services

Web services programming
- SOAP Web services
- REST Web services

History of creating application

Programming with

- 0 and 1
- Assembly
- Procedural programming language
- OOP programming
- Service-Oriented Architecture

What is a service?

Restaurant **provides** food: a service

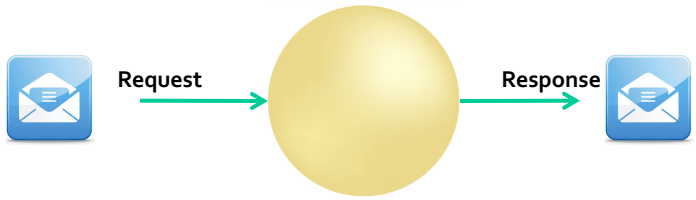
After the order is taken, food is produced, served, …: service may **consist** of other services

The menu indicates the service provided: a service **description**

The order is written down, or yelled at, the cook: services **communicate** through **messages**

What is a service in IT?

An entity that provides some capability to its clients by exchanging messages (request - response)

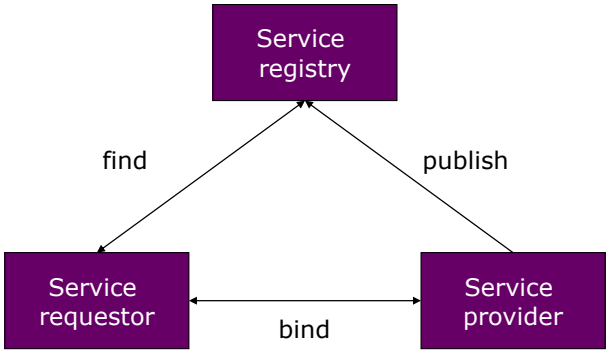


Service characteristics

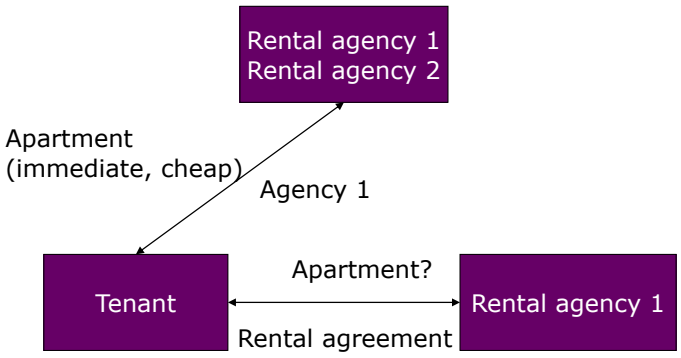
- Services can (must) be **discovered**
- Services can be **composed** to form larger services
- Services adhere to a service **description/contract**
- Services are **loosely coupled**
- Services are **stateless**

- Services are **autonomous**
- Services **hide** their logic
- Services are **reusable**
- Services use **open standards**

Service discovered



Example



Loosely coupled, stateless, autonomous & hide logic

- Loosely coupled: system is one in which each of its components has, or makes use of, little or no knowledge of the definitions of other separate components
- Stateless: Rental agencies come and go, the agencies cannot retain information: it doesn't know if and when it will be invoked again, and by whom
- Autonomous: Its logic does not depend on the tenants
- Hide logic: Rental agency has its own rules on how to structure its process

What is SOA?

SOA is a software architecture model in which business functionality are logically grouped and encapsulated into self contained, distinct and reusable units called **services**

represent a **high level** business concept
can be **distributed** over a network
can be **reused** to create new business applications
contain **contract** with specification of the purpose, functionality, interfaces, usage

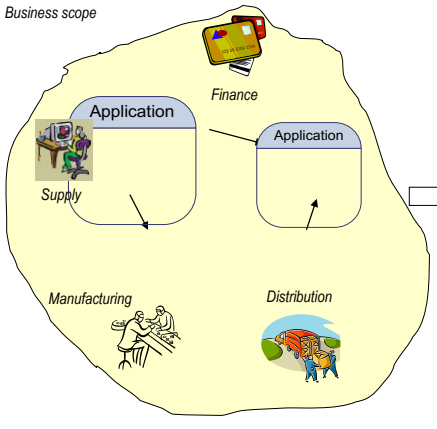
SOA from different views

A set of **business services** composed to capture the *business design* that an enterprise wants to expose to its customers and clients ↔ **Business**

An **architectural style** that requires a service provider, requester and a service description ↔ **Architecture**

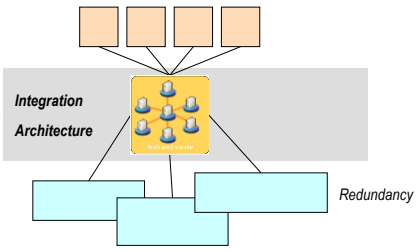
A **programming model** complete with standards, tools, methods and technologies such as Web services ↔ **Programming**

Application centric



Business functionality is duplicated in each application that requires it.

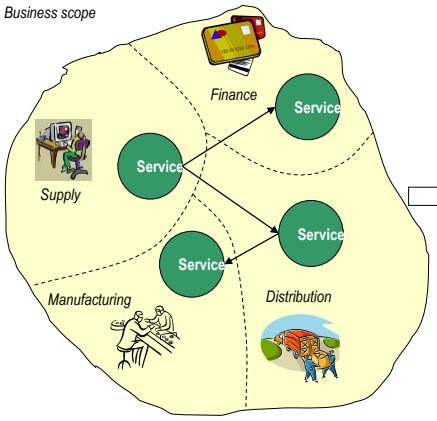
Narrow Consumers
Limited Business Processes



Redundancy

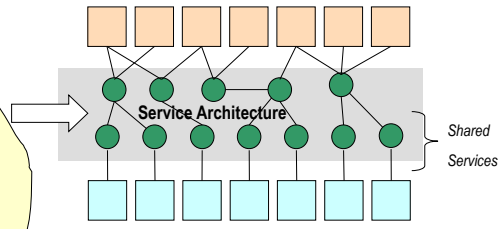
Overlapped resources
Overlapped providers

Service centric



SOA structures the business and its systems as a set of capabilities that are offered as Services, organized into a Service Architecture

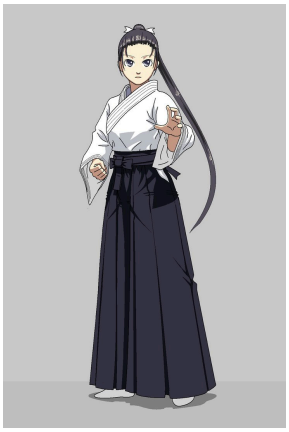
Multiple Service Consumers
Multiple Business Processes



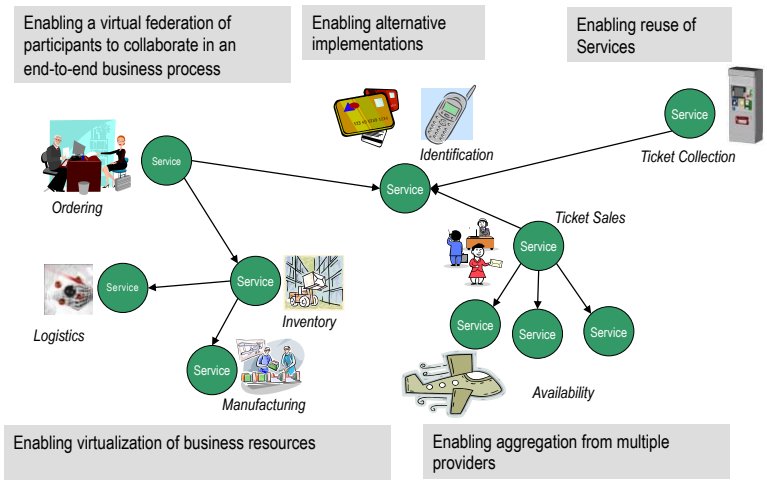
Multiple Discrete Resources
Multiple Service Providers

Service virtualizes how that capability is performed, and where and by whom the resources are provided, enabling multiple providers and consumers to participate together in shared business activities.

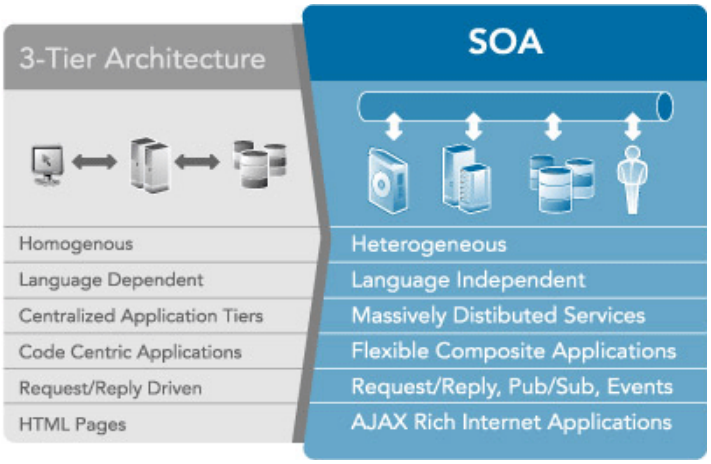
Application centric vs Service centric



Why SOA?



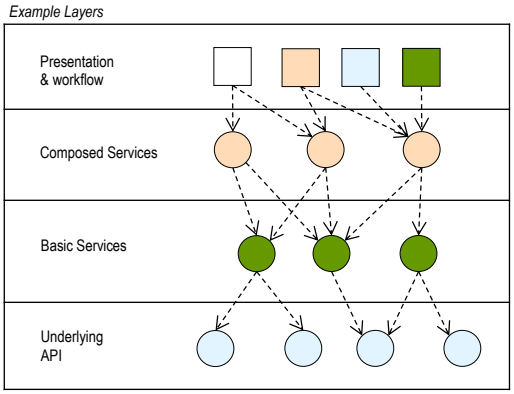
SOA is an evolutionary step



SOA layered architecture

Reasons for Layering

1. **Flexible composition**.
2. **Reuse**.
3. **Functional standardization** in lower levels
4. **Customization** in higher layers
5. **Separation of Concerns.**
6. **Policies may vary by Layer**



TOPICS

Introduction to SOA
- Service
- Service Oriented Architecture

Web services

Web services programming
- SOAP Web services
- REST Web services

Web services & SOA

Web services is chosen for SOA because:

- Loose coupling
  Service requesters depend only the interface described in WSDL and not on the implementation of the service provide
- Interoperability
  Service interactions are based on the exchange of XML-based SOAP messages over standards based transport protocols
- Wide industry support
  Wide industry support for adoption of Web services standards promotes interoperability of various vendor platforms that support Web services

What is web service?

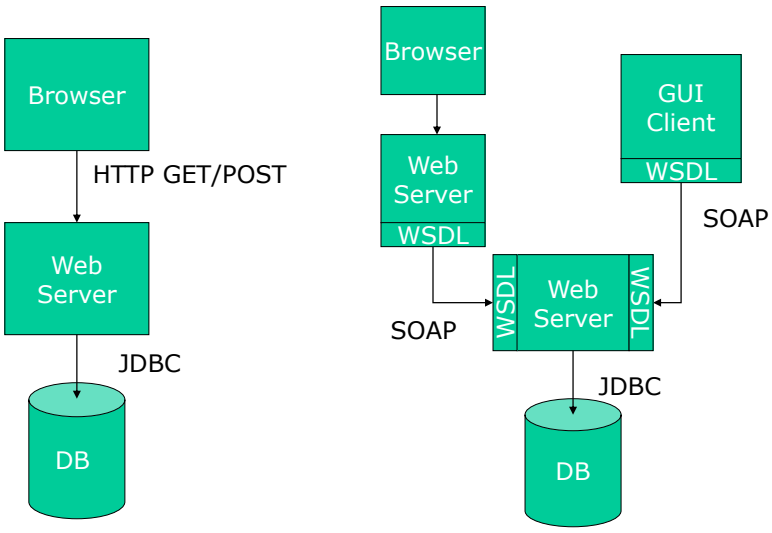
Implementation means to realize services

Based on open standards:

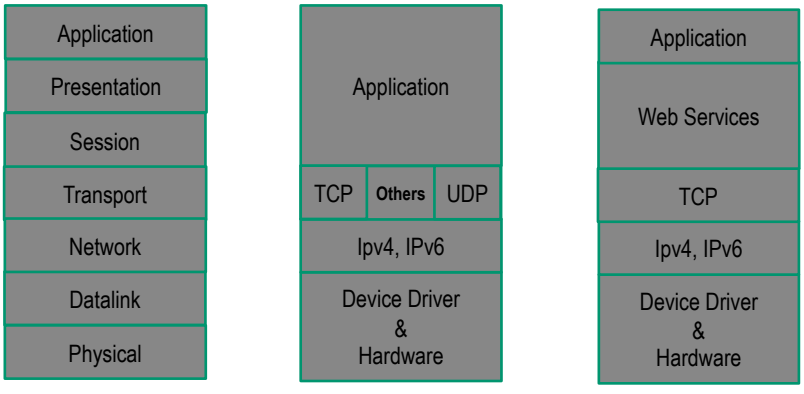
- XML
- SOAP: Simple Object Access Protocol
- WSDL: Web Services Description Language
- UDDI: Universal Description, Discovery and Integration
- BPEL4WS: Business Process Execution Language for Web Services

Main standardization bodies: OASIS, W3C

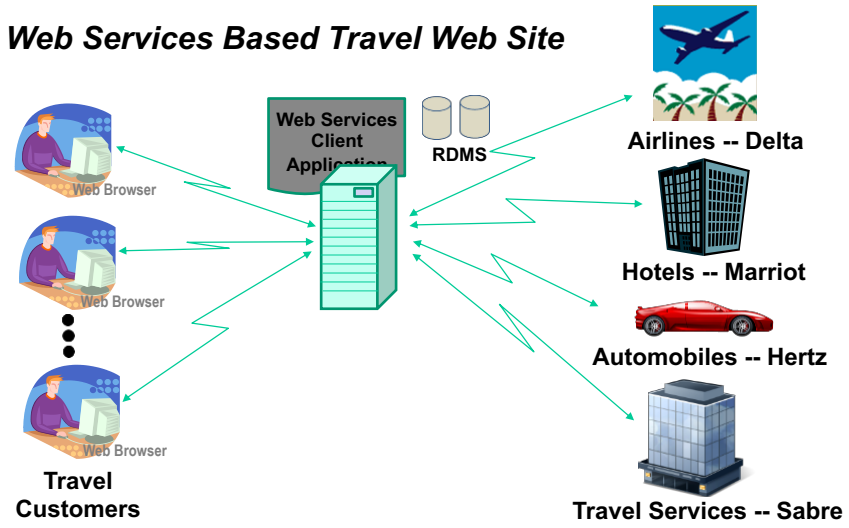
One for all …



OSI and Web service



Example



Open standard technologies for web service

XML – tagging data such that it can be exchanged between applications and platforms

SOAP – messaging protocol for transporting information and instructions between applications (uses XML)

WSDL – a standard method of describing web services and their specific capabilities (XML)

UDDI – defines XML-based rules for building directories in which companies advertise themselves and their web services

XML

Developed from Standard Generalized Markup Method (SGML)
XML widely supported by W3C
Essential characteristic is the separation of content from presentation
XML describes only data
Any application that understands XML can exchange data

XML

XML parser checks syntax
If syntax is good the document is *well-formed*
XML document can optionally reference a *Document Type Definition (DTD),* also called a *Schema*
If an XML document adheres to the structure of the schema it is *valid*

SOAP

SOAP enables between distributed systems

SOAP message has three parts

- *envelope* – wraps entire message and contains header and body
- *header* – optional element with additional info such as security or routing
- *body* – application-specific data being commuicated

WSDL

Web services are self-describing

Description is written in WSDL, an XML-based language through which a web service conveys to applications the methods that the service provides and how those methods are accessed

WSDL is meant to be read by applications (not humans)

UDDI

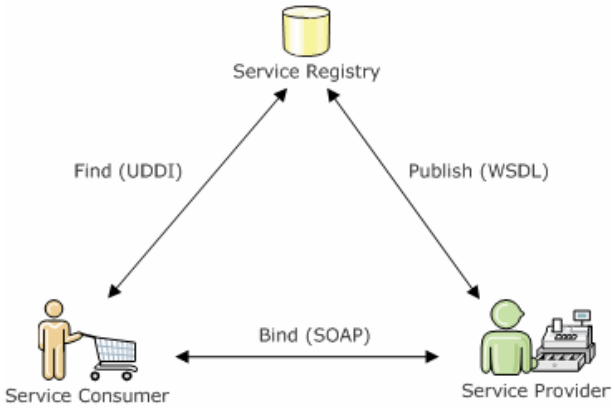
UDDI defines an XML-based format that describes electronic capabilities and business processes

Entries are stored in a UDDI registry

UDDI Business Registry (UBR)

- "white pages" – contact info, description
- "yellow pages" – classification info, details
- "green pages" – technical data

Web service in SOA



TOPICS

Introduction to SOA
- Service
- Service Oriented Architecture

Web services

Web services programming
- SOAP Web services
- REST Web services

Three Most Common Styles of Use

RPC (Remote Procedure Calls)
- A distributed function call interface

SOAP (Simple Object Access Protocol)
- The basic unit of communication is a message, rather than an operation

REST (Representational State Transfer)
- Standard operations in HTTP: GET, POST, PUT, DELETE
- Interacting with stateful resources, rather than messages or operations

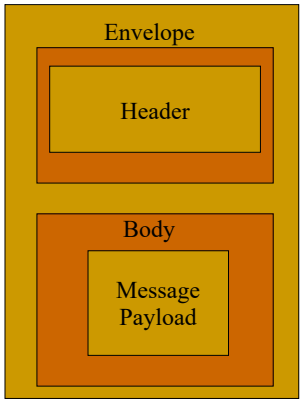
RPC Web Services

Basic unit: WSDL operation
Widely deployed and supported, but not loosely coupled
Other approaches: CORBA, DCE/RPC, Java RMI

SOAP Web Services

Basic unit: message

Supported by most major vendors, loose coupling



Representational State Transfer (REST)

Interacting with stateful resources, rather than messages or operations

Using HTTP standard operations such as GET, POST, PUT, DELETE

WSDL 2.0 offers support for binding to all HTTP request methods

- WSDL 1.1 only GET and POST

SOAP web service

- HTTP-XML-based protocol
- Enables application to communicate over Internet
- Uses XML documents called messages
- SOAP message contains an envelope
- Describes message's content and intended recipient
- Ability to make a Remote Procedure Call (RPC)
- Request to another machine to run a task

SOAP

- Using Web Services and SOAP, the request would look something like this:

```
<?xml version="1.0"?>
<soap:Envelope
  xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
   soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
        <soap:body pb="http://www.acme.com/phonebook">
                <pb:GetUserDetails>
                        <pb:UserID>12345</pb:UserID>
                </pb:GetUserDetails>
        </soap:Body>
</soap:Envelope>
```

Pros & cons

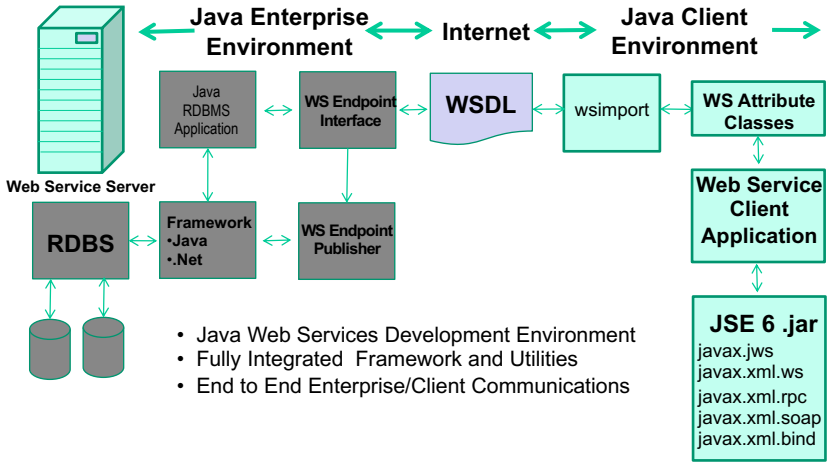
Advantages

- Human readable XML
- Easy to debug
- SOAP runs over HTTP
- Firewalls not affected
- Services can be written in any language, platform or operating system

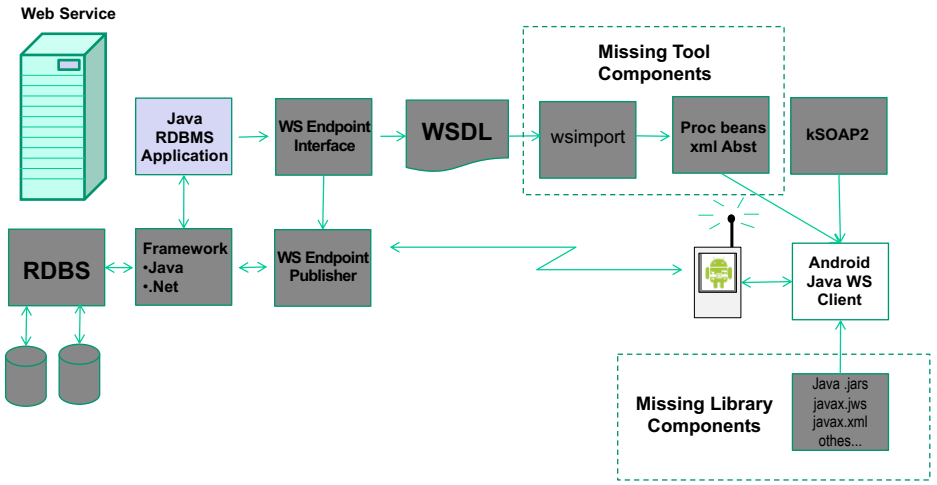
Disadvantages

- S-L-O……..-W
- XML produces a lot of overhead for small messages
- Web Services speed relies on Internet traffic conditions
- Not strictly-typed XML

SOAP web service by Java



Android Web Services Development Environment



SOAP vs REST

SOAP	REST	
Meaning	Simple Object Access Protocol	Representational State Transfer
Design	**Standardized protocol with pre-defined rules to follow.**	**Architectural style with loose guidelines and recommendations.**
Approach	Function-driven (data available as services, e.g.: "getUser")	Data-driven (data available as resources, e.g. "user").
Statefulness	**Stateless by default, but it's possible to make a SOAP API stateful.**	**Stateless (no server-side sessions).**
Caching	API calls cannot be cached.	API calls can be cached.
Security	**WS-Security with SSL support. Built-in ACID compliance.**	**Supports HTTPS and SSL.**
Performance	Requires more bandwidth and computing power.	Requires fewer resources.
Message format	**Only XML.**	**Plain text, HTML, XML, JSON, YAML, and others.**
Transfer protocol(s)	HTTP, SMTP, UDP, and others.	Only HTTP
Recommended for	**Enterprise apps, high-security apps, distributed environment, financial services, payment gateways, telecommunication services.**	**Public APIs for web services, mobile services, social networks.**
Advantages	High security, standardized, extensibility.	Scalability, better performance, browser-friendliness, flexibility.
Disadvantages	**Poorer performance, more complexity, less flexibility.**	**Less security, not suitable for distributed environments.**

email: chungdt@soict.hust.edu.vn

Q&A