WEB SERVICE AND SOA FUNDAMENTALS

Examples

- Self-contained business task such as "funds withdrawal or funds deposit service"
- Full-fledged business process, such as automated "bill inquiry handling"
- An application, such as a "life insurance application or demand forecasts and stock replenishment"
- A service enabled resource, such as "access to a particular back end database containing patient record"

WEB SERVICES

- Simple requests to complete business applications that access and combine information from multiple sources
    - Credit checking and authorization
    - Pricing inquiries
    - Inventory status checking
    - Weather report
- a distributed technology that enables disparate applications running on different machines to exchange messages and integrate with one another without requiring additional, proprietary third party software or hardware.
- Self-describing and self-contained network available software modules that perform concrete tasks and are deployed easily because they are based on common industry standards and existing technology, such as XML and HTTP.
- Reduce application interface costs, and provide a universal mechanism for integrating business processes within an enterprise and, ultimately, among multiple organizations.

What problems do Web Services are addressing?

It addresses the rigid implementations of predefined relationships and isolated services scattered across the internet.

SERVICE ORIENTED COMPUTING

- A computing paradigm that utilizes services as the constructs to support the development of rapid, low cost composition of distributed applications.
- Services are self-contained modules deployed over standard middleware platforms that can be described published located orchestrated and programmed using xml-based

technologies over a network any piece of code and any application component developed on a system can be transformed into a network available service

- Software services (or simply **services**) a service-oriented Approach to programming , based on the idea of describing available computational resources , for example application programs and information system components , as services that can be delivered through a standard and well-defined interface.
- Service-oriented computing represents the fusion of several technologies including distributed systems, software engineering, information systems, Computer languages , web-based computing , and xml technologies , rather than being a new technology

"An application can no longer be thought of as a single process running within a single organization. The value of an application is actually no longer measured by its functional capabilities but rather by its ability to integrate with its surrounding environment"

Papazoglou 2003

What does the quote above mean?

Services can help integrate applications that were not written with the intent to be integrated with other applications and define architectures and to build new functionality leveraging existing application functionality.

Composite applications - a collection of interactive services offering well defined interfaces to their potential users. These applications process the ability to integrate with other service based applications.

Loosely coupled applications - this relationship is used by services, it is the link between applications of transacting partners. At the middleware level, this concept requires that the service-oriented approach be independent of specific technologies or OS. The service oriented model does not even mandate any kind of predetermined agreements before the use of an offered service is allowed.

THREE TYPES OF SOFTWARE ORGANIZATIONS

1. Service Providers - provide the service implementations, supply their service descriptions and provide related technical and business support
2. Service Clients - end user or consumer organizations that use some service
3. Service Aggregators - organizations that consolidate multiple services into a new, single orchestrated service offering that is commonly known as a business process

A major advantage of services is that they may be implemented on a single machine or on a large number of variety of devices and be distributed on a local area network or more widely across several wide-area networks including mobile and ad-hoc networks

SOFTWARE AS A SERVICE (SaaS)

- Application service providers (ASP) are companies that package software and infrastructure elements together with business & professional services to create a complete solution that they present to the end customer as service on a subscription basis. This model is also called *on-demand software* or *software as a service (SaaS)*
- Basic idea behind this model is to bill customers on a per-use basis or rent applications to subscribers on a monthly/annual fee.

Is it good or bad for the customers or for the service provider?

- ☐ Access the applications remotely through a web browser
- ☐ Outsource some or even all aspects of IT needs
- ☐ Small opportunity to customize the application
- ☐ Limited control of functionalities
- ☐ Relies on the provider for customization
- ☐ The provider takes primary responsibility for managing the software application on its infrastructure
- ☐ The provider maintains the application the infrastructure customer data and ensures that the systems and data are available whenever needed

WEB SERVICES VS WEB BASED APPLICATIONS

When comparing Web services to Web-based applications we may distinguish four key differences [Aldrich 2002]:

- Web services act as resources to other applications that can request and initiate those Web services, with or without human intervention. This means that Web services can call on other Web services to outsource parts of a complex transaction to those other Web services. This provides a high degree of flexibility and adaptability not available in today's Web-based applications.
- Web services are modular, self-aware, and self-describing applications; a Web service knows what functions it can perform and what inputs it requires to produce its outputs, and can describe this to potential users and to other Web services. A Web service can also describe its *non-functional properties*: for instance, the cost of invoking the service, the geographical areas the Web service covers, security measures involved in using the Web service, performance characteristics, contact information, and more (see section 1.8).
- Web services are more visible and manageable than Web-based applications; the state of a Web service can be monitored and managed at any time by using external application management and workflow systems. Despite the fact that a Web service may not run on an in-house (local) system or may be written in an unfamiliar programming language, it still can be used by local applications, which may detect its state (active or available) and manage the status of its outcome.
- Web services may be brokered or auctioned. If several Web services perform the same task, then several applications may place bids for the opportunity to use the requested service. A broker can base its choice on the attributes of the "competing" Web services (cost, speed, degree of security).

What is the difference between Web services and a Web based application?

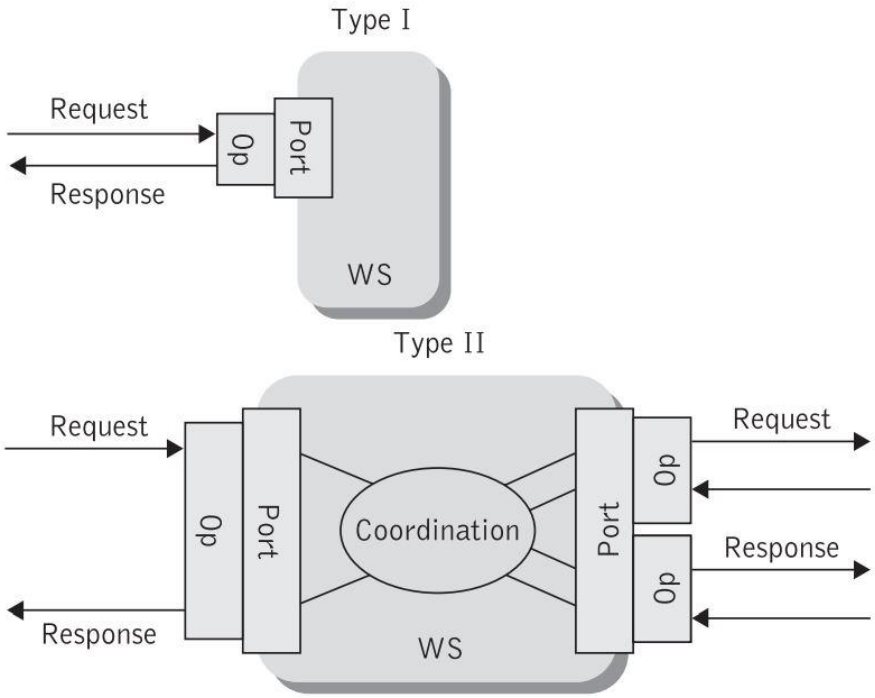
CHARACTERISTICS OF WEB SERVICES

Important characteristics of Web services (simple, complex web services, stateful, stateless services, service granularity, loose coupling, synchronous, asynchronous, etc)

Types of Web service

I. **Simple or Informational** (atomic or discrete or singular) - provide Programmatic access to content , interacting with an end-user by means of simple request or response sequences or alternatively may expose back and business applications to other applications

II. **Complex Services or business process** - Involved the assembly and indication of many pre-existing services , possibly found in diverse enterprises , to complete a multi-step business interaction that requires coordination



: High-level view of informational and complex services

Sub-categories of Informational Services

1. Pure content services
    a. Weather report information
    b. Simple financial information
    c. Stock quote information
    d. Design information
    e. News item
2. Simple trading services - can provide a seamless aggregation of information across disparate existing systems and information sources, including back end systems, giving programmatic access to a business information system so that the requester can make informed decisions
    a. Logistic services, where automated services are the actual front ends to fairly complex physical organizational IS

3. Simple syndication services - value-added information web services that purport to plug into commerce sites of various types, such as e-marketplaces, or sell-sites. These are services that are offered by a third party and run the whole range from commerce enabling services, such as logistics, payment fulfillment, and tracking services, to other value-added commerce services, such as rating services.
    a. Reservation services on a travel site
    b. Rate quote services on an insurance site

Two Types of Complex Web Services

1. **Complex services that compose programmatic Web services** - the clients of these web services can assemble them to build complex services
    a. Ex. inventory checking service that comprises part of an inventory management process
2. **Complex services that compose interactive Web services** - these services expose the functionality of a web application's presentation (browser) layer. They frequently expose a multi-step Web application behavior that combines a Web server, an application server and underlying database systems, and typically deliver the application directly to a browser and eventually to a human user for interaction

FUNCTIONAL AND NON-FUNCTIONAL PROPERTIES

Services are described in terms of a description language

Two Major Interrelated Components

1. **Functional Description** - the operational characteristics that define the overall behavior of the service. It focuses on the operational details regarding the syntax of messages and how to configure the network protocol to deliver messages
    a. How the service is invoked
    b. The location where it is invoked and so on
2. **Non-functional Description** - concentrates on the service quality attributes
    a. Service metering and cost
    b. Performance metrics
        i. Response time
        ii. Accuracy
    c. Security attributes
        i. Authorization
        ii. Authentication
    d. (Transactional) Integrity
    e. Reliability
    f. Scalability
    g. Availability

STATE PROPERTIES

Services could be stateless or stateful.

- **Stateless** - if services can be invoked repeatedly without having to maintain context or state
- **Stateful** - if services that may require their context to be preserved from one invocation to the next

Connectionless protocol

The service access protocol is always connectionless. The connectionless protocol means that the protocol has no concepts of a job or a session and does not make any assumptions about eventual delivery.

Examples

1. Informational weather report service
2. Stateful order management service

LOOSE COUPLING

Tightly coupled message exchange applications

- need to know how their partner applications behave.
- need to know intimate details of how their partner requires to be communicated with
    - The number of methods it exposes and the details of the parameters that each method accepts, and the type of results it returns
- Need to know the location of the applications with which they work (implies security guarantee)
- It is brittle because when any form of change or replacement is required to parts of the whole application
- Difficult and cumbersome to build because, when the number of applications and services increases, the number of interfaces that need to be created and maintained quickly becomes unwieldy

Loosely coupled message exchange

- need not know or care how their partner applications behave or are implemented
- Agile and could survive evolutionary changes in the structure and implementation of the internals of each service, which make up the whole application
- Asynchronous or event driven model rather than a synchronous model of interaction
- Provides a level of flexibility and interoperability that cannot be matched using traditional approaches to building highly integrated, cross-platform, program-to-program communications environments
- Service requester has no knowledge of the technical details of the provider's implementation
    - Ex programming language, deployment platform, etc.
- Service requester typically invokes operations by way of messages - a request message and the response - rather than through the use of API

Tight versus loose coupling

	TIGHT COUPLING	LOOSE COUPLING
Method of communication	Synchronous	Asynchronous
Messaging Style	RPC-style	Document-style
Message path	Hard coded	Routed
Underlying platform	Homogenous	Heterogenous
Binding protocol	Static	Dynamic - late binding
Overall objective	Re use	Re use, flexibility, broad applicability

SERVICE GRANULARITY

- Web services vary in function from simple requests to complex systems that access and combine information from multiple sources.
- Simple services are discrete in nature, exhibit normally a request/reply mode of operation, and are of fine granularity
- Complex services are coarse grained. Coarse grained communication implies larger and richer data structures
    - Ex. supports XML schema, enables looser coupling, enables asynchronous communication where the information exchange is the minimum required to complete the task

SERVICE SYNCHRONICITY

Two Methods of Communication or Programming Styles

➔ **Synchronous services** *or Remote Procedure Call (RPC)*
- ◆ Clients express their request as a method call with a set of arguments, which returns a response containing a return value.
- ◆ When a client sends a request message, it expects a response message before continuing with its computation
- ◆ “ALL-or-nothing proposition”
- ◆ If one operation is unable to complete for any reason, all other dependent operations will fail (tightly-coupled model of communication between client and service provider)
- ◆ When to use RPC-style web services
    - The client invoking the service requires immediate response
    - The client and service work in a back and forth conversational way

➔ Asynchronous services
- ◆ Document-style or message driven services

- ◆ Client sends a message as the entire document, such as a purchase order, rather than a discrete set of parameters.
- ◆ The service accepts the entire document; processes it and may or may not return a result message.
- ◆ A client that invokes async service does not need to wait for a response before it continues with the remainder of its application.
- ◆ The response can appear hours or even days later
- ◆ An application does not need to know the immediate details of how to reach and interface with other applications
- ◆ Allows a communication operation between any two processes to be a self-contained, standalone unit of work.
- ◆ When to use Document style web services?
    - The client does not require (or expect) an immediate response
    - The service is document oriented. This means that the client typically sends an entire document

WELL-DEFINEDNESS

The service interaction must be well defined using a standard definition language format. Applications follow the standard rules for interfacing and interacting. A standard definition language provides a uniform mechanism for describing abstract service interfaces and specific protocol bindings that support the service.

SERVICE USAGE CONTEXT

It is also important to divide services into different categories based on the web service requester's perspective

- Replaceable Web service
    - A service provided by several providers, and replacing one provider with another does not affect application functionality as long as the service interfaces are identical
    - The productivity is not impacted severely if the service is unavailable for a short period of time as another provider (and service) may be chosen.
- Mission-critical web service
    - A service possibly provided by a single specific provider, which if replaced compromises severely the functionality of an entire application.
    - If the service were unavailable for a period of time it would drastically reduce the productivity of the application. This type of service would typically hold some critical business data and be integrated at the process level