Web Services

Description: A Web service is a loosely coupled, encapsulated, platform and programming language neutral, composable server-side component that can be described, published, discovered and invoked over an internal network or on the Internet.

- Web services are based on a collection of XML standards for building Distributed Systems in heterogeneous environments
- Key characteristic – Web Services process XML formatted SOAP messages

Comparison with DS

- WS can be assessed in a more ad-hoc manner, enabling them to be more easily used in Internet applications
- The URI of a Web Service can be compared with the remote object reference
- SOAP is similar to traditional binary protocols used in CORBA and RMI, but instead of binary format they adapt a text based XML data representation
- XML textual representations takes more space than binary and require more time to process

Main Differences from Distributed Object Model are

- Remote objects are not instantiated
- Garbage collection is irrelevant
- Remote Object References are irrelevant

Web Service Architectural Layers

Service-Oriented Architecture (SOA) is an approach to distributed computing that considers software resources as services available on the network.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Standards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications</td>
<td>Java, .NET</td>
</tr>
<tr>
<td>Transactions / Security</td>
<td>WS-Transaction / WS-Security</td>
</tr>
<tr>
<td>Collaboration, Coordination</td>
<td>WS-BPEL, WS-CDL</td>
</tr>
<tr>
<td>Discovery, Directory</td>
<td>UDDI, WSDL</td>
</tr>
<tr>
<td>Service Descriptions</td>
<td>WSDL</td>
</tr>
<tr>
<td>Messaging</td>
<td>SOAP/XML, REST</td>
</tr>
<tr>
<td>Transport Protocol</td>
<td>HTTP, HTTPS, FTP, SMTP</td>
</tr>
</tbody>
</table>
XML

- XML is a de facto standard for structuring data, content and data format for electronic documents
- XML is a textual representation, although bulky has been adopted for its readability
- External data representation and marshalling of messages exchanged between Web services is done in XML
- XML Schema: describes the syntax and context and constraints of XML documents
- XML Processing:
  - DOM – The XML file can be viewed and processed as a tree
  - SAX – Scans the document from top to bottom (event based)

SOAP

- SOAP is a stateless communication protocol
- It is designed to support loosely coupled applications that interact by exchanging messages
- SOAP is used to encapsulate messages and transmit them over HTTP or other protocol (SMTP, TCP, UDP)
- SOAP assumes that every message has a sender, an ultimate receiver and number of intermediaries
- SOAP is based on the packaging of single one way messages. It supports request reply interactions by using pairs of single messages.
- SOAP can be used for synchronous and asynchronous communications

SOAP

The message structure

- **Envelope**: Container of the message’s structure
- **Header**: Optional part. Provide information necessary for intermediate processing or added value services (Security)
- **Body**: Mandatory part. Encapsulates the actual message
- **Fault**: To handle errors, or status information

Example – SOAP

Two aspects influence how the header and body of soap are constructed:

- The interaction style (Document or RPC)
- Encoding rules (SOAP encoding - defines how basic and complex data structures are serialised)

WSDL

WSDL: Provides an interface enabling clients to interact with servers in a general way

- WSDL is separated into abstract and concrete part
- **Abstract Part**: Is analogous to conventional IDLs
  - Port Type definitions (Interfaces). Each Port type is a collection of operations
  - Operations: Specify message exchange pattern. Available options are (in-out, in-only, out-in, out-only, Robust In only, Robust Out Only)
  - Messages: abstract definition of the data being transmitted
  - Types: XML Data type definitions used to describe messages

WSDL

**Concrete Part**: Defines the protocols bindings and address information

- **Binding (How)**: Defines message format and protocol details. soap:binding element has two attributes - the style [rpc or document], and the transport such as SOAP over HTTP
  - For each operation a corresponding SOAP action is defined. We also specify how the input and output are encoded [literal or encoded]
- **Service (Where)**: Specifies the service location and one or more ports (locations) where an instance of the service can be contacted
Another common need in middleware is for naming or directory service to allow clients to find out about services. UDDI: Provides a framework for describing publishing and discovering Web Services. It provides a sophisticated naming and directory service. Provides an API for looking up services and publication.

UDDI contains 4 types of information:
- Business Entity: Describes the organisation that provides the Web Service such as company's name, address etc.
- Business Service: Represents the service offered by a company.
- Binding Template: Instructions on how to invoke a service, its address, information about the service interfaces.
- Model: "Technical Model", holds service descriptions such as WSDL documents.

Basic Web services implement simple interactions. Many applications require coordinated (particular order) sequences of interactions. In RPCs, adding guarantees to the interactions / transactions require additional protocol (two phase commit) and supporting infrastructure (TC monitors). However, Web service activities may take long, so would be impractical to use two phase commit protocol. There is a need for a more relaxed protocol. WS-Coordination: defines a coordination protocol, infrastructure to provide transactional semantics for WS.

Web Service Composition:
- Composition is the task of combining and linking existing Web Services and other components to create new services or processes.
- WS-BPEL: Defines executable process specification defining the implementation logic of a composite service.
- Take a service centric perspective.
- External Web services – are participants associated with specific roles.
- Defines constructs for: variables, exceptions, timeouts, synchronous and a synchronous interactions, parallel and repetitive execution etc.

Securing Web Services:
XML security consists of a set of related standards for encryption, signing and key management:
- Data encryption at different granularity levels (unlike to TLS).
- Canonical XML: any two logically equivalent XML documents have the same canonical form and are considered identical.

Technologies Behind securing Web Services:
- XML – ENC: Specifies the way to represent encrypted data in XML and the process of encrypting and decrypting.
- XML DSIG: Provides a mechanism for applying digital signatures to XML documents and other resources.
- XKMS: It provides a protocol for distributing and registering public keys for use XML signatures.
**Semantic Web Services**

- XML/WSDL is not enough as a self-describing language.
- Semantic Web enables the dynamic discovery/collaboration of Web services through standardized semantic descriptions of the functionality of a service.
- Semantic Web Services is based on a common (shared) understanding of a given domain and the concepts and tasks relevant to this domain.
- Ontology: formally defines objects and their properties (classes, relations) of the domain.
- DAML-S provides an upper ontology for describing properties and capabilities of Web services in an unambiguous, computer interpretable markup language.

**Grid**

- What is the Grid? - Resource sharing & coordinated problem solving.
- Grid: Provides a uniform computing environment for scientists and engineers to solve data computational intensive problems.
- Middleware, designed to enable sharing of resources among group of users.
- Example – Astronomy / Genome / Biology Communities
- Management and Coordination of Resources is needed.
- Run over Web Services.

---

**OGSA (Open Grid Service Architecture)** – a standard for building service-oriented Grid systems.

- OGSA – Introduces the concept of **Grid Services** which are Web Services with some extensions to meet the specific needs of Grid.

- OGSA - extensions introducing interfaces and conventions in **three main areas**.
  - First, there is a need for interfaces to manage the creation, destruction and life cycle management of services.
  - Second, there is state. Grid services can have attributes and data associated with them.
  - Third, clients can subscribe their interests in services. Once there is any change in a service, the clients are notified. This is a call-back operation from services to clients.

**Exercise**