Chapter 19. Web Services

- Introduction

- **Web Services**
  - XML
  - SOAP

- Service Descriptions

- A directory service for use with web services

- XML security

- Coordination of web services
19.1 Introduction

- **Web service**: is the name for a method or function that is available for other applications to access over the Internet

- **Web server vs. web services**
  - Web server: provides a basic HTTP service
  - Web service: provides a service based on the operations defined in its interface

- The provision of web services as an addition to web services is based on the ability to use an HTTP request to cause the execution of a program
  - HTTP request-reply protocol, allows general-purpose clients called browsers to view web pages and other resources with reference to their URLs
  - When a URL in an HTTP request refers to an executable program, the result is produced by that program and returned.

- External data representation and marshalling of messages exchanged between clients and web services is done in XML

- **SOAP protocol** specifies the rules for using XML to package messages
A web service is identified by a URI and can be accessed by clients using messages formatted in XML.

SOAP is used to encapsulate these messages and transmit them over HTTP.

A web service deploys service descriptions (interface definition, server’s URL info. etc.) to specify the interface and other aspects of the service for the benefit of potential clients.

Web services and applications may be built on top of other web services.

Some particular web services provide general functionality required for the operation of a large number of other web services.

- **Choreography**: means for coordinating operations of web services.

<table>
<thead>
<tr>
<th>Applications</th>
<th>Directory service</th>
<th>Security</th>
<th>Choreography</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Services</td>
<td>Service descriptions (in WSDL)</td>
<td>SOAP</td>
<td>URIs (URLs or URNs)</td>
</tr>
</tbody>
</table>
URI, URL, and URN

- URI (Uniform Resource Identifier): a general resource identifier, whose value may be either URL or URN
- URL: includes resource location information
- URN (Uniform Resource Names): location independent, rely on lookup service to map them onto the URLs of resources
19.2 Web services

- A web service interface generally consists of a collection of operations that can be used by a client over the internet.

- The key characteristic of most web services is they can process XML-formatted SOAP messages.

![Diagram showing the process of encoding and decoding between C++, SOAP, UTF-8, and Java.]
XML

- Page 150 (4th edition)
- XML is a markup language defined by the World Wide Web Consortium (W3C)
- Both XML and HTML were derived from SGML (Standardized Generalized Markup Language)
- Use of tags:
  - HTML: used to specify how a browser could display the text
  - XML: used to describe the logical structure of the data
- XML is extensible in the sense that users can define their own tags, in contrast to HTML, which uses a fixed set of tags
- XML defines a textual format for representing structured data
  - Originally intended for documents containing textual self-describing structured data
  - Now also used to represent the data sent in messages exchanged by clients and servers in web services
- XML in web service: clients usually use SOAP messages to communicate with web services.
  - SOAP message is in XML format whose tags are published for use by web services and their clients
- Readable but large. However, files and messages can be compressed
XML definition of the Person structure

<person id="123456789">
  <name>Smith</name>
  <place>London</place>
  <year>1934</year>
  <!-- a comment -->
</person>

- **Element**: `<name>Smith</name>`
- **Attribute**: `id="123456789"`
- **items represented as elements or attributes**:
  - An element is generally a container for data
  - An attribute is used for labelling that data
XML definition of the Person structure

- **XML document must be well-formed**
  - Conform to rules about its structure
  - i.e. each start tag has a matching end tag,
  - all tags are correctly nested. E.g. `<x>..<y>…</x>..<y>` is incorrect

- **XML prolog**: each XML document must have a prolog as its first line, which must at least specify the version in use (currently 1.0). May also specify the encoding (UTF-8 by default), standalone or dependent on external definitions

  ```xml
  <?XML version = "1.0" encoding = "UTF-8" standalone = "yes"?>
  ```

- **DTD (Document Type Definition)**: definitions of entities and specifications that tell which tags are valid in the document.
  - Can be defined directly within the prolog, as well as pointers to external specification files.
  - Can be checked to see everything within tags adheres to the limitations imposed by the DTD (Validation)
  - Not used for defining web services, but may be used to define documents transmitted by web services

- **Stylability**: stylesheet standard, XSL, lets you dictate how to portray the data
Illustration of the use of a namespace in the Person structure

```
<person pers:id="123456789" xmlns:pers = "http://www.cdk4.net/person">
  <pers:name> Smith </pers:name>
  <pers:place> London </pers:place>
  <pers:year> 1934 </pers:year>
</person>
```

- XML namespaces: a set of names for a collection of element types and attributes, that is referenced by a URL
- Any element that makes use of an XML namespace can specify that namespace as an attribute called xmlns, whose value is a URL referring to the file containing the namespace definitions
- In the example, prefix pers is bound to http://www.cdk4.net/person
An XML schema for the Person structure

```xml
<xsd:schema xmlns:xsd = "URL of XML schema definitions" >

  <xsd:element name = "person" type = "personType" />

  <xsd:complexType name = "personType">
    <xsd:sequence>
      <xsd:element name = "name"  type = "xs:string"/>
      <xsd:element name = "place"  type = "xs:string"/>
      <xsd:element name = "year"  type = "xs:positiveInteger"/>
    </xsd:sequence>
    <xsd:attribute name = "id"   type = "xs:positiveInteger"/>
  </xsd:complexType>

</xsd:schema>
```

- An XML schema defines the elements and attributes that can appear in a document, how the element are nested and the number of elements, whether an element is empty or can include text.
- For each element, it defines the type and default value
- A single schema definition may be shared by many different documents.
- An XML document that is defined to conform to a particular schema may also be validated by means of that schema. E.g. sender of a SOAP message may use an XML schema to encode it and the recipient will use the same XML schema to validate and decode it
SOAP (Simple Object Access Protocol)

- SOAP is a XML-based lightweight protocol for exchange of information in a decentralized, distributed environment
  - It defines a scheme for using XML to represent the contents of request and reply messages
  - Platform independent, language independent
- Originally was based only on HTTP, but current version can use other transport protocols including SMTP, TCP or UDP
- To support client-server communication, SOAP specifies how to use the HTTP POST method for the request message and its response for the reply message. The combined use of XML and HTTP provides a standard protocol for client-server communication over the Internet.
- SOAP APIs have been implemented in many programming languages, including Java, Javascript, Perl, Python, .NET, C, C++, C#, and VB
- Programmers do not normally need to concern how SOAP uses XML to represent messages and HTTP to communicate them
A web service using SOAP

In this scenario, your application wants some information that lives on a remote computer

1. your application sends out a request for data, and the first stop on its journey is the SOAP client, which might also live on your computer
2. this soap client takes in the request for data and translate the parameters into a SOAP message
3. the client then sends this SOAP message to the remote computer, which has a SOAP server running at all times, listening for incoming SOAP messages
4. The SOAP server listens for SOAP messages, take the input data in them, and translate it into something the remote method can understand
5. It then passes the input data to the remote method
6. the method does its thing and comes up with some data

... ...
Aside from the HTTP-specific data, the XML document contains three parts specified by XML elements:

- **Envelope**: `<SOAP-ENV:Envelope>`
  The SOAP envelope is analogous to a snail mail envelope, but without the address, which is the responsibility of the transport and included in the HTTP header. The envelope specifies global settings such as the encoding.

- **Header**: `<SOAP-ENV:Header>`
  The header is optional. If it is present, it contains header entries that define SOAP settings, such as the ultimate destination of a message and application-specific settings (the transaction identifier, for instance).

- **Body**: `<SOAP-ENV:Body>`
  The body must be present and must follow the header, if any. The body contains either a Request or a Reply.
Example of a simple request without headers

env:envelope xmlns:env = namespace URI for SOAP envelopes

env:body

m:exchange
xmlns:m = namespace URI of the service description

m:arg1 Hello
m:arg2 World

- The body encloses and element with the name of the procedure to be called and the URI of the namespace (the file containing the XML schema) for the relevant service description, denoted by m.
- Inner elements contain the arguments of the procedure.
Corresponding successful reply message

- Two output arguments
- The name of the procedure has “Response” added to it
- Uses same two XML schemas as the request message:
  - The first defining the SOAP envelope, and second procedure and argument names
Comparisons

- With RMI: superficially similar, RMI client uses a remote object reference to invoke an operation; web service client uses a URI
- With CORBA: CORBA was designed for use within a single organization or between a small number of collaborating organizations

Ease of use:
- HTTP and XML infrastructure for web services is well-understood and convenient to use and is already installed on all of the most commonly used operating systems, although the user does require a convenient programming language API to SOAP.
- In contrast, CORBA platform is a large and complex piece of software requiring installation and support.

Efficiency:
- CORBA has been designed to be efficient: CORBA CDR is binary, whereas XML is textual
- Study shows SOAP request messages are 14 times as large as the equivalent ones in CORBA and that a SOAP request took 882 times as long as an equivalent CORBA invocation.
- But the performance difference is not noticeable in some applications
- W3C has been investigating the possibility of allowing binary data to be included in XML elements so as to increase efficiency
19.3 Service descriptions

- Interface definitions are needed to allow clients to communicate with services.
- For web services, interface definitions are provided as part of a more general service description, which specifies two other additional characteristics:
  - How the messages are to be communicated (SOAP or HTTP)
  - The URI of the service
- In the web service context, Web Services Description Language (WSDL) is commonly used for service descriptions.
- It defines an XML schema for representing the components of a service description, separates the abstract part from the concrete part.

The main elements in a WSDL description: types, message, interface, bindings, services.
19.4 A directory service for use with web services

How can clients obtain service descriptions?

- Search services of that type and come across a web page advertising the service
- Use a directory service
- Any organization that plans to base its applications on web services will find it more convenient to use a directory service to make these services available to clients.
- Universal Directory and Discovery Service (UDDI): provides both a name service and a directory service. That is, WSDL service descriptions may be looked up by name (a white pages service) or by attribute (a yellow pages service). They may also be accessed directly via URLs
- Clients may use yellow pages approach to look up a particular category, or use white pages approach to look up a service with reference to the organization providing it
- Data structures: designed to support all the above styles of access
  - businessEntity: name, address etc of the organization
  - businessServices: category information
  - bindingTemplate: holds address of a web service instance and references to service descriptions
  - tModel: holds service descriptions, usually WSDL documents, stored outside the database and accessed by means of URLs
The main UDDI data structures

- **businessEntity**: human readable information about the publisher
  - **businessServices**: human readable information about a family of services
    - **bindingTemplate**: information about the key service interfaces
      - **key**
      - **URL**
    - **bindingTemplate**
  - **key**

- **tModel**: service descriptions
  - **tModel**
  - **URL**
19.5 XML security

• XML security consists of a set of related W3C designs for signing, key management and encryption

• Intended for use in cooperative work over the Internet involving documents whose contents may need to be authenticated or encrypted

• Typically the documents are created, exchanged, stored and then exchanged again, possibly after modifications by a series of different users

• E.g. a document containing a patient’s medical records: Different parts would be created/modified by doctors/nurses/consultants/administrators/pharmacists
  – Different parts will be viewable by different roles

• These needs cannot be met by TLS (Transport Layer Security protocol), which is previously known as SSL (Secure Sockets Layer protocol)
  – TLS is supported by most browsers and is widely used in Internet commerce
  – Used to create a secure channel for the communication of information

• The security must be specified within the document itself and applied to the document rather than as a property of the channel
  – This is possible in XML, in which metadata can be used
Requirements

- **Basic requirements**: XML security should provide at least the same level of protection as TLS, that is
  - To be able to encrypt either an entire document or just selected parts of it
  - To be able to sign either an entire document or just selected parts of it

- **Additional basic requirements**: arise from the need to store documents, possibly to modify them and then to send them on to different recipients
  - To add to a document that is already signed and to sign the result
  - To add to a document that already contains encrypted sections and to encrypt part of the new version, possibly including some of the already encrypted sections
  - To authorize various different users to view different parts of a document

- **Requirements concerning algorithms** …

- **Requirements for finding keys** …
Other issues

- Canonical XML
- Use of digital signatures in XML
- Key management service
- XML encryption

Project:
- SOAP and XML?
- and XML security?
19.6 & 19.7 Two terms

- **Choreography**: a language based on WSDL for defining coordination of web services.
  - E.g. might specify constraints on the order and the conditions in which messages are exchanged by participants
  - Intended to provide a global description of a set of interactions
  - SOAP infrastructure supports single request-response interactions
  - However, many useful applications involve several requests that need to be done in a particular order

- **Grid**: middleware that is designed to enable the sharing of resources such as files, computers, software, data and sensors on a very large scale
  - Resources (data or computer power) are shared typically by groups of users in different organizations who are collaborating on the solution of problems requiring large number of computers
Summary

- Infrastructure of web services:
  - widely-used HTTP to transport messages (based on URIs to refer to resources)
  - XML (a textual format) for data representation and marshalling

- SOAP: the communication protocol generally used by web services & clients
  - Request or reply SOAP message is enclosed in XML-formatted document called envelope

- Conventional middleware uses interface definitions to provide clients with the details of services. In case of web services, service descriptions are used, commonly written in WSDL, and specify:
  - Communication protocol to be used, e.g. SOAP
  - URI of the service
  - Describing its interface

- XML security was designed to provide the necessary protection for the contents of a document exchanged by members of a group of people, who have different tasks to perform on that document.