Web Services for Mobile Intelligent Devices

Dr. Ghodrat Moghadampour
mg@puv.fi
VAMK, University of Applied Sciences
Vaasa, Finland

VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Outline

- Web Today & Future
- Web applications
- Web services
- XMLRPC
- SOAP
- J2ME Wireless API

Web, Today &Future

- Human-centric Web:
  - most of today's Web, where humans are the main actors initiating most web requests, like e-commerce
  - the customer sends requests and gets the response as an HTML page

Web, Today & Future

- Application-Centric Web :
  - humans are not entirely out of the picture
  - the conversations can take place directly between applications as easily as between web browsers and servers
  - web services can automatically take care of functionalities, like querying the database and showing the results when other applications and agents connect to them.

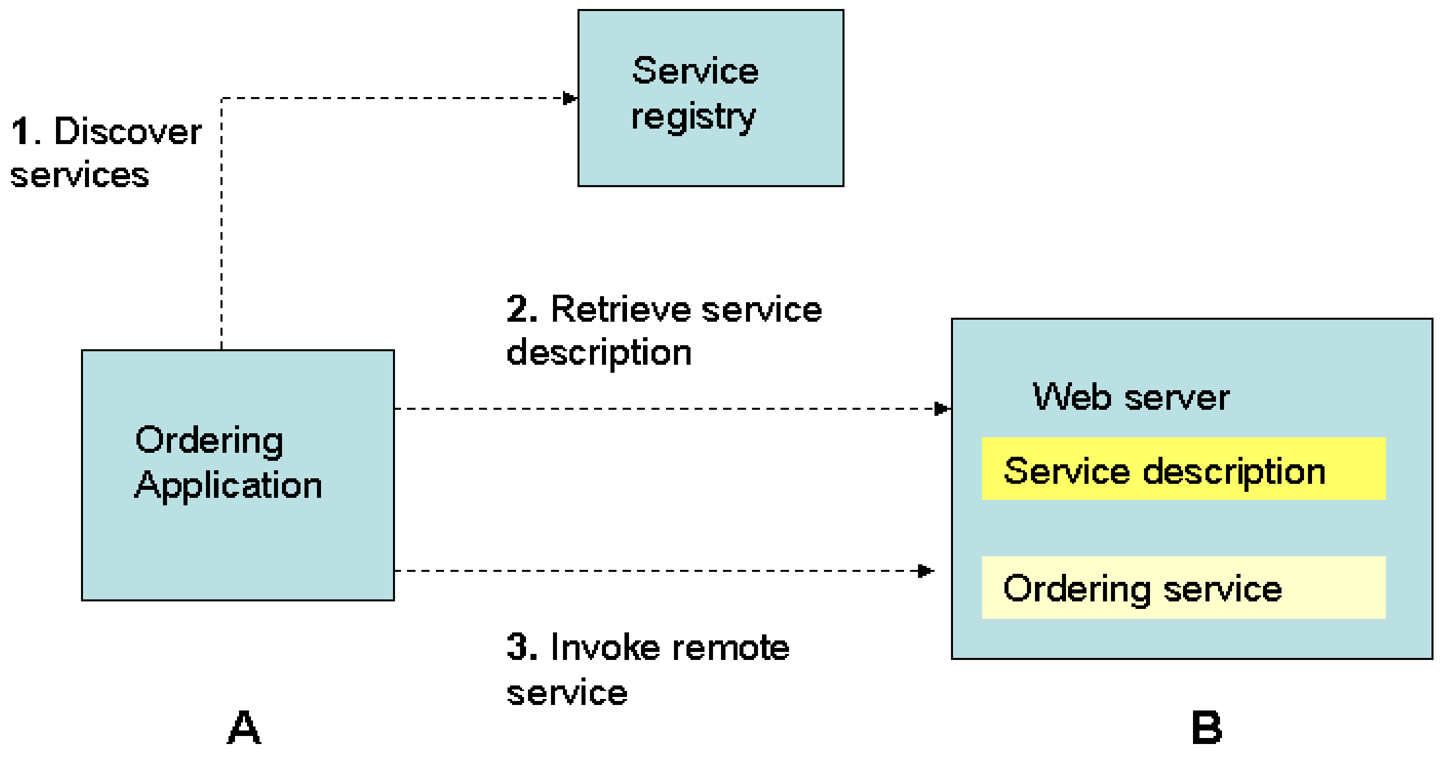
Web, Today & Future

- Automated Web:
  - the application-centric Web is not a new idea: credit card services, search systems, etc.
  - however, so far these systems have been consisted of rather ad hoc than standard solutions
  - If web services are easily discoverable, self-describing and stick to common standards, it is possible to automate, or "just-in-time" application integration

Web, Today & Future

- Just-in-time application integration:
  1. application A connects to a centralized directory of web services and asks whether application B supports ordering services
  2. the directory returns information on available services and includes a pointer to the ordering service description.
  3. application A connects to application B and retrieves the service description

Web Today & Future



Web Applications

- Server side applications:
  - communicate through Internet, HTTP, HTTPS, FTP, SMTP, …
  - one standard client application (Web browser)
  - do not need to take into account the needs of various client applications
  - do not need to communicate with client applications via a common standard language

Web Applications

- Client side application:
  - is web browser
  - GUI is implemented using HTML, DHTML, XML, JavaScripts, …
  - data received from the server is typically only displayed on the client side

Web Services

- Server side applications:
  - communicate through Internet, HTTP, HTTPS, SMTP, FTP, BEEP
  - must take into account the needs of various applications running on different operating systems
  - must communicate through a standard language (XML)

Web Services

- Client side applications:
  - user-made applications
  - not standard
  - do not need to be written in a certain language
  - do not nee to be running on a specific operating system
  - must be capable of communicating with a common language (XML)

Web Services

- Optional and desirable properties:
  - Self-describing:
    - the web service should have a public interface
  - Discoverability:
    - there should be a relatively simple mechanism to discover the web service and locate its public interface; a completely decentralized system or a more logically centralized registry system

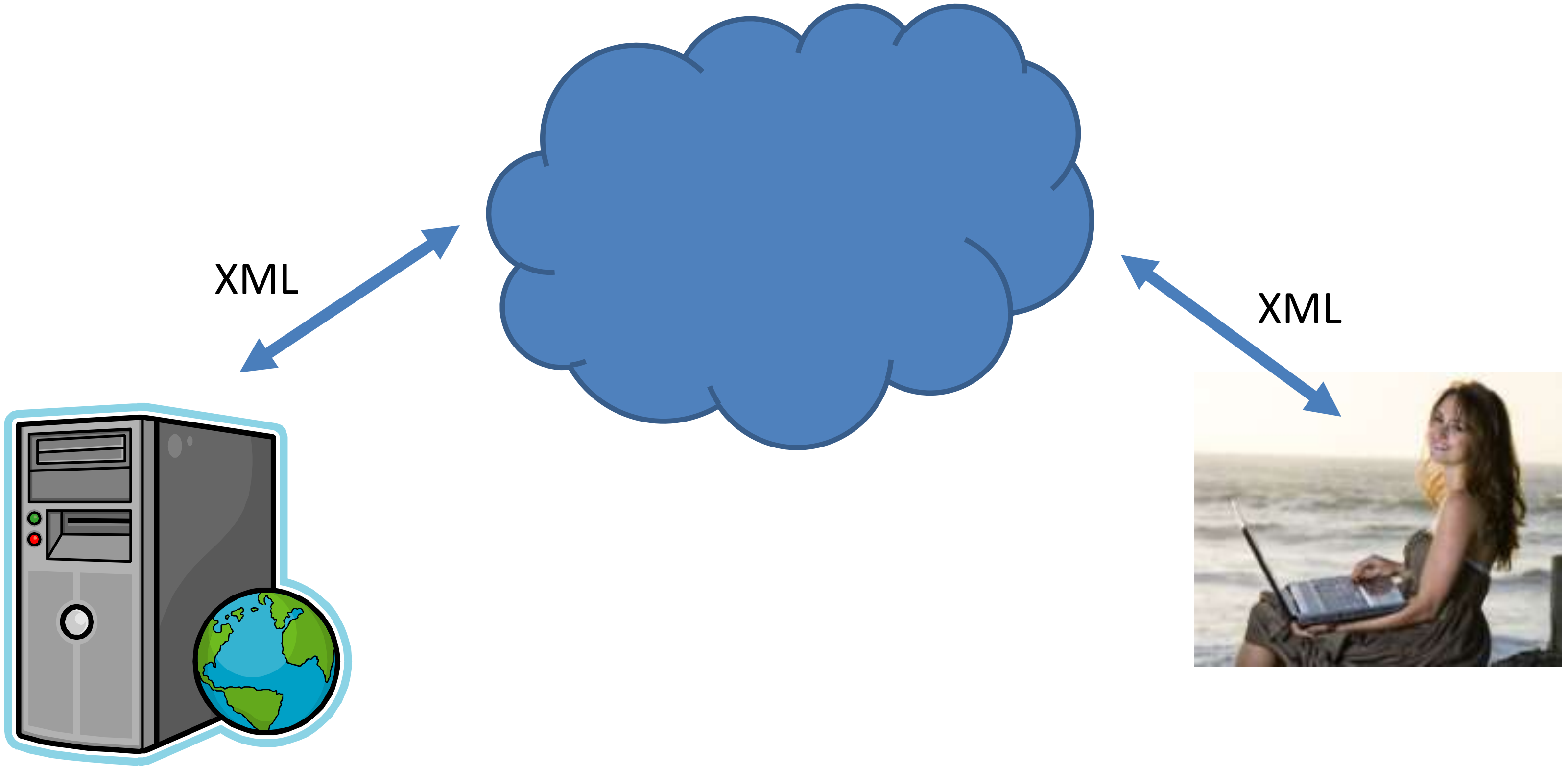
XML Messaging

- HTTP GET/POST methods
- XML Remote Procedure Calls (XML-RPC):
  - a simple protocol which uses XML to encode its calls and HTTP POST as a transport mechanism. XML responses are embedded in the body of the HTTP response.
  - it is platform-independent and allows diverse applications to communicate; a C# client can speak XML-RPC to a Java server.

XML Messaging

- Simple Object Access Protocol (SOAP):
  - an XML-based protocol for exchanging information between computers
  - can be used in a variety of messaging systems and can be delivered via a variety of transport protocols
  - the main focus is RPCs transported via HTTP
  - platform-independent and; thus enables diverse applications to communicate

Web Services



Web Service Protocol Stack

- **Service transport**:
  - responsible for transporting messages between applications, like Hypertext transfer protocol (HTTP), Simple Mail Transfer Protocol (SMTP), file transfer protocol (FTP) and newer protocols, such as Blocks Extensible Exchange Protocol (BEEP), which is a peer to peer network protocol

Web Service Protocol Stack

- **XML messaging**:
  - responsible for encoding messages in a common XML format so that messages can be understood at either end
  - This layer includes XML-RPC and SOAP
- **Service description**:
  - responsible for describing the public interface to a specific web service.

Web Service Protocol Stack

  - This is currently done using the Web Service Description language (WSDL)
- **Service directory**:
  - responsible for centralizing services into a common registry, and providing easy publish/find functionality
  - This is currently handled via Universal Description, Discovery and Integration (UDDI)

Web Service Protocol Stack

Discovery	UDDI
Description	WSDL
XML Messaging	XML-RPC, SOAP, XML
Transport	HTTP, SMTP, FTP, BEEP

J2ME Web Services API

- The WSA extends the Java 2 Platform, Micro Edition to support web services.
- The API provides two standardized packages, which are crucial to clients of web services:
  - remote service invocation
  - XML parsing

J2ME Web Services API

- WSA is designed to work with J2ME profiles based on either the Connected Device Configuration (CDC) or the Connected Limited Device Configuration (CLDC 1.o or CLDC 1.1).
- The remote invocation API is based on a strict subset of J2SE's Java API for XML-Based RPC (JAX-RPC 1.1), with some Remote Method Invocation (RMI) classes included to satisfy JAX-RPC dependencies.

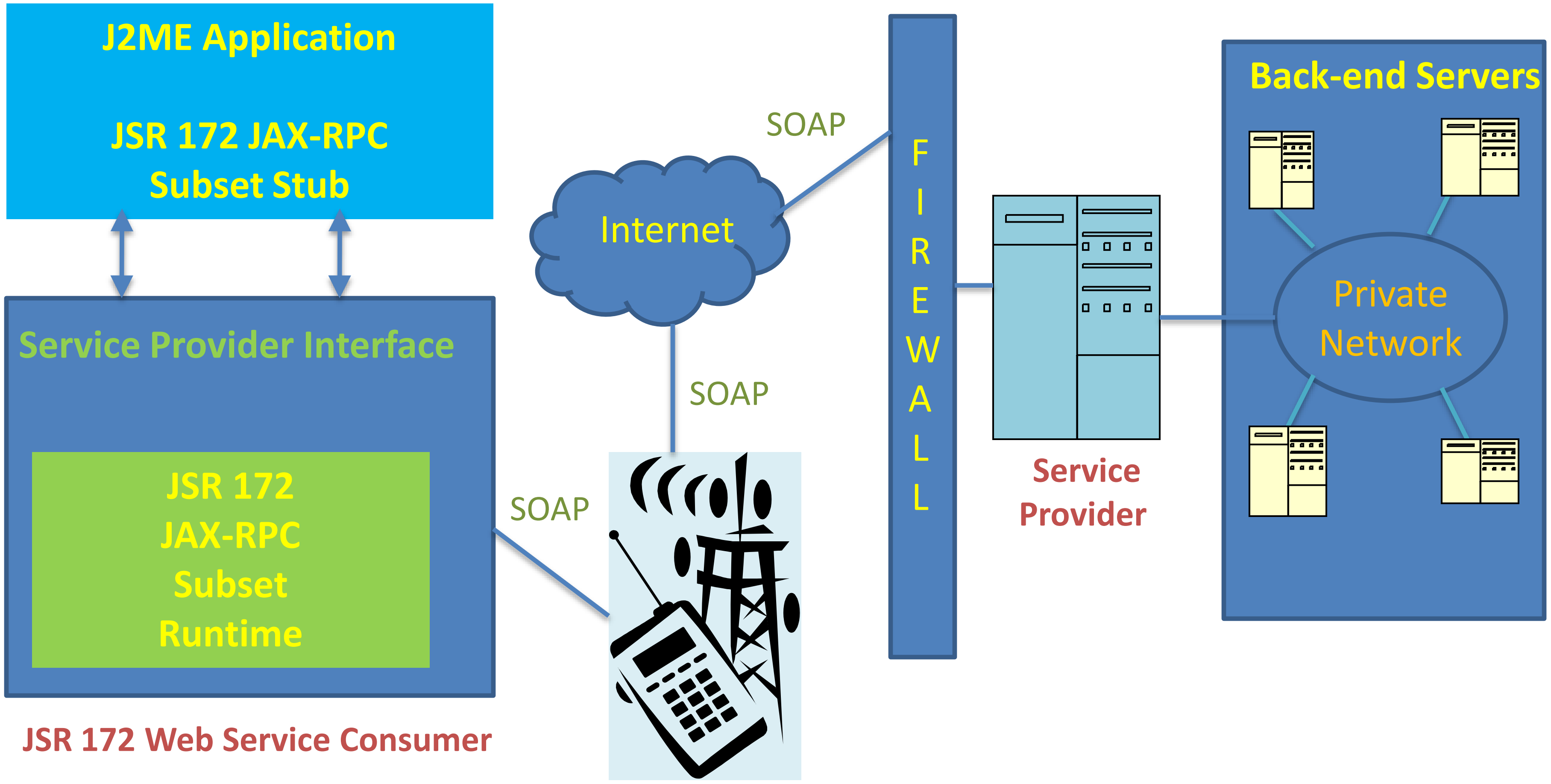
J2ME Web Services API

- The XML-parsing API is based on a strict subset of the Simple API for XML, version 2 (SAX2)
- WSA provides fundamental support for web services invocation and XML parsing into the device's runtime environment
- Reduces memory requirements on mobile devices

J2ME Web Services API

- JSR 172 specifies standardized client-side technology to enable J2ME applications to consume remote services on typical web services architectures:

J2ME Web Service Communication


J2ME Application
JSR 172 JAX-RPC Subset Stub
Service Provider Interface
JSR 172 JAX-RPC Subset Runtime
JSR 172 Web Service Consumer
SOAP
Internet
SOAP
SOAP
FIREWALL
Service Provider
Back-end Servers
Private Network

J2ME Web Services API

- This web service architecture has three elements:
  - A network-aware application residing on a WSA-enabled wireless device. The application includes a JSR 172 stub that uses the JSR 172 runtime to communicate with the network
  - The wireless network and the Internet, and the corresponding communication and data-encoding protocols, including binary protocols, HTTP, and SOAP/XML

J2ME Web Services API

– A web server, acting as the service producer, typically behind one or more firewalls and a proxy gateway. The web server often provides access to back-end applications and servers on a private network.

J2ME Web Services API

- The application itself is a smart client based on the Mobile Information Device Profile (MIDP) or the Personal Basis Profile (PBP), with business-specific logic, user interface, persistence logic, and life-cycle and application-state management
- To handle XML documents, the application can employ the JAXP subset API
- To consume web services, it can use the JAX-RPC subset API, employing JSR 172 stubs and the runtime

J2ME Web Services API

- In devices such as cell phones, typically the application and the JSR 172 stub reside in the device's memory, while all the JSR 172 elements, along with the underlying profile and configuration, are embedded in the device itself.

J2ME Web Services API

- At the center of JSR 172 operations is the runtime, with its service provider interface, which enables the stubs to perform all the tasks associated with invoking an RPC service endpoint:
  - Set properties specific to an RPC invocation
  - Describe the RPC invocation input and return values

J2ME Web Services API

- Encode input values
- Invoke the RPC service endpoint
- Decode and return to the application any values that the service endpoint returns

J2ME Web Services API

- J2ME Web services are based on the general Web services architecture and specifications, but looking at them from strictly the client or service consumer, J2ME Web services consists of two optional packages:
  - Remote service invocation (Java API for XML-based RPC or JAX-RPC)
  - XML parsing (Java API for XML Processing or JAXP)

Thank you… ☺