มหาวิทยาลัยขอนแก่น
วิทยา จริยา ปัญญา KHON KAEN UNIVERSITY

Simple Object Access Protocol (SOAP)

Asst. Prof. Dr. Kanda Runapongsa Saikaew
Department of Computer Engineering
Khon Kaen University
http://gear.kku.ac.th/~krunapon/xmlws

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น
FACULTY OF ENGINEERING KHON KAEN UNIVERSITY
1
1

Agenda

- What is and What is Not SOAP?
- SOAP Message Structure
- SOAP Terminology
- SOAP Message Exchange
- Document vs. RPC

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น
FACULTY OF ENGINEERING KHON KAEN UNIVERSITY
2
2

What is SOAP? (1/3)

- SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment
- SOAP uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols
- The framework has been designed to be independent of any particular programming model and other implementation specific semantics

What is SOAP? (2/3)

- Simple Object Access Protocol
- Wire protocol similar to
  - IIOP for CORBA
  - JRMP for RMI
- XML is used for data encoding
  - "text" based protocol vs. "binary" protocol
- Supports XML-based RPC (Remote Procedure Call)

What is SOAP? (3/3)

- Stateless
- One-way message exchange paradigm
  - Applications can create more complex interaction patterns (e.g., request/response, etc.) by combining several messages
- Silent on the semantics of any application-specific data it conveys
- Does not specify the details of many issues, such as the routing of SOAP messages, etc.
- But provides a full description of the required actions taken by a SOAP node on receiving a SOAP message

What SOAP is Not

- Not a component model
  - So it will not replace objects and components, i.e., EJB, JavaBeans
- Not a programming language
  - So it will not replace Java
- Not a solution for all
  - So it will not replace other distributed computing technologies such as RMI

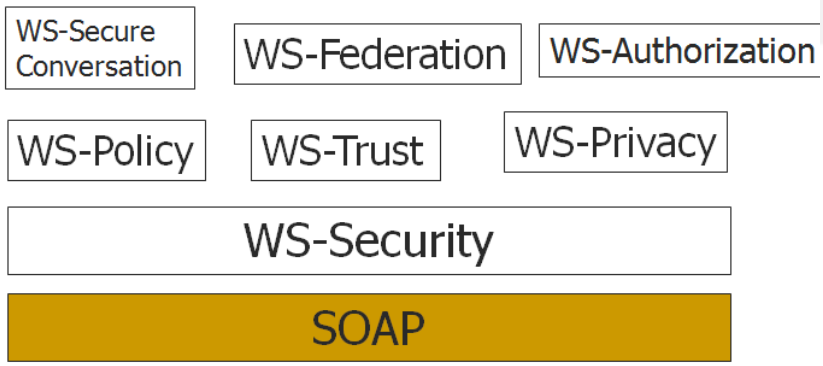
Why Do We Need SOAP?

- An acceptable standard for routing and packaging XML data exchanged between two applications on a network
- No need to define our own networking, addressing, and routing protocols

SOAP Design Goals

- Simplicity
- Extensibility
  - New standards define new semantics
- Features not supported (by design)
  - Distributed garbage collection
  - Object by reference
  - Activation
  - Message batching

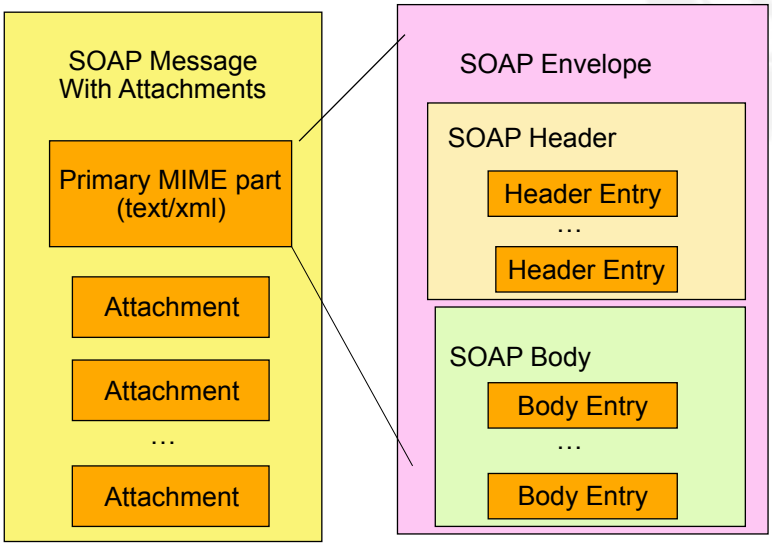
SOAP is the foundation



Agenda

- What is and What is Not SOAP?
- SOAP Message Structure
- SOAP Terminology
- SOAP Message Exchange
- Document vs. RPC

SOAP Message Format



SOAP Message Envelope

- Embedded Information
  - Namespaces
  - Encoding information
- Header
  - Optional
  - Can be handled by intermediaries
- Body
  - Mandatory
  - Handled only by ultimate receiver

SOAP Header (<env:Header>)

- Used for extension
  - Context
  - Authentication
  - Transaction
  - Management
- Made of Header blocks (Header entries)
- Most Web services standard activities are basically defining standard header entries for a particular domain

SOAP Header Blocks

- Child elements of SOAP Header
- Designed in anticipation of various uses for SOAP by SOAP intermediaries
  - Can be individually targeted at SOAP nodes
  - Allows SOAP intermediaries to provide value-added services
- May be inspected, inserted, deleted or forwarded by SOAP nodes encountered along a SOAP message path

SOAP Body (<env:Body>)

- Made of Body blocks
- Consumed by Ultimate SOAP receiver
- Carry main end-to-end information
  - Application data (XML document) (document style)
  - RPC method and parameters (rpc style)
  - SOAP fault

SOAP Fault (<env:Fault>)

- Used to carry error and/or status information
- Four sub-elements
  - faultcode
  - faultstring
  - faultactor
  - detail

Pre-defined SOAP Faultcode Values

- VersionMismatch
  - Invalid namespace in SOAP envelope
- MustUnderstand
  - Receiver mode cannot handle *mustUnderstand* SOAP header block
- Client
  - Indicates client side error
- Server
  - Indicates server side error

SOAP Fault Example: Cause

```
<env:Envelope
    xmlns:env='http://www.w3.org/2001/06/soap-envelope'>
    <env:Header>
        <abc:Extension xmlns:abc='http://example.org/2004/10/ext'
    env:mustUnderstand='1'/>
    </env:Header>
    <env:Body>
        …
    </env:Body>
</env:Envelope>
```

SOAP Fault Example: Result

```
<env:Envelope xmlns:env='http://www.w3.org/2001/06/
soap-envelope' xmlns:f='http://www.w3.org/2001/06/
soap-faults'>
    <env:Header>
        <f:Misunderstood qname='abc:Extension'/>
    </env:Header>
    <env:Body>
        <env:Fault>
            <faultcode>MustUnderstand</faultcode>
            <faultstring>One or more mandatory headers
not understood</faultstring>
    …
</env:Envelope>
```


Header Block or Body Block?

- Decision made at the time of application design
- Header blocks may be targeted at various nodes that might be encountered along a message's path from a sender to the ultimate recipient
  - Intermediate SOAP nodes may provide value-added services based on data in such headers

Agenda

- What is and What is Not SOAP?
- SOAP Message Structure
- SOAP Terminology
- SOAP Message Exchange
- Document vs. RPC

Message Sender & Receiver Concepts

- SOAP sender
  - A SOAP node that transmits a SOAP message
- SOAP receiver
  - A SOAP node that accepts a SOAP message
- SOAP message path
  - The set of SOAP nodes through which a single SOAP message passes
  - This includes the initial SOAP sender, zero or more SOAP intermediaries, and an ultimate SOAP receiver

Message Sender & Receiver Concepts

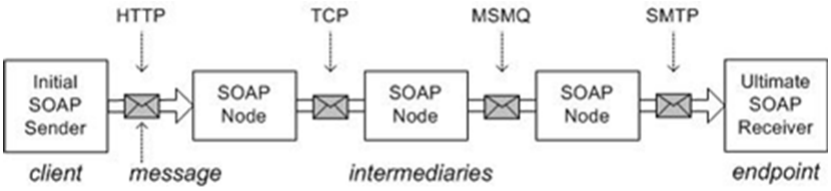
- Initial SOAP sender
  - The SOAP sender that originates a SOAP message at the starting point of a SOAP message path
- SOAP intermediary
  - A SOAP intermediary is both a SOAP receiver and a SOAP sender and is targetable from within a SOAP message
  - It processes the SOAP header blocks targeted at it and acts to forward a SOAP message towards an ultimate SOAP receiver

Message Sender & Receiver Concepts

- Ultimate SOAP receiver
  - The SOAP receiver that is a final destination of a SOAP message
  - It is responsible for processing the contents of the SOAP body and any SOAP header blocks targeted at it
  - An ultimate SOAP receiver cannot also be a SOAP intermediary for the same SOAP message

SOAP over Transport Protocol

- SOAP can be used over any transport protocol such as TCP, HTTP, SMTP



Agenda

- What is and What is Not SOAP?
- SOAP Message Structure
- SOAP Terminology
- SOAP Message Exchange
- Document vs. RPC

Example: SOAP RPC Request (1/2)

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
    <env:Header>
        <t:transaction
        xmlns:t="http://thirdparty.example.org/transaction"
        env:encodingStyle="http://example.com/encoding"
        env:mustUnderstand="true" >5</t:transaction>
</env:Header>
```


Example: SOAP RPC Request (2/2)

```
<env:Body>
    <m:chargeReservation
    env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
    xmlns:m="http://travelcompany.example.org/">
    <m:reservation
        xmlns:m="http://travelcompany.example.org/reservation">
        <m:code>FT35ZBQ</m:code>
    </m:reservation>
    <o:creditCard xmlns:o="http://mycompany.example.com/financial">
        <n:name xmlns:n="http://mycompany.example.com/employees">
        Åke Jógvan Øyvind
        </n:name>
        <o:number>123456789099999</o:number>
        <o:expiration>2005-02</o:expiration>
        </o:creditCard>
    </m:chargeReservation>
    </env:Body>
</env:Envelope>
```

Example : RPC Response (1/2)

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
    <env:Header>
        <t:transaction
        xmlns:t="http://thirdparty.example.org/transaction"
        env:encodingStyle="http://example.com/encoding"
        env:mustUnderstand="true">5</t:transaction>
    </env:Header>
    <env:Body>
        <m:chargeReservationResponse
            env:encodingStyle="http://www.w3.org/2003/05/soap-
    encoding"
            xmlns:m="http://travelcompany.example.org/">
            <m:code>FT35ZBQ</m:code>
        <m:viewAt>
        http://travelcompany.example.org/reservations?code=FT35ZBQ
        </m:viewAt>
        </m:chargeReservationResponse>
    </env:Body>
</env:Envelope>
```


Example : RPC Response (2/2)

```
<?xml version='1.0' ?>
    <env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
    <env:Header>
        <t:transaction
        xmlns:t="http://thirdparty.example.org/transaction"
        env:encodingStyle="http://example.com/encoding"
        env:mustUnderstand="true">5</t:transaction>
    </env:Header>
    <env:Body>
        <m:chargeReservationResponse
        env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
        xmlns:rpc="http://www.w3.org/2003/05/soap-rpc"
        xmlns:m="http://travelcompany.example.org/">
            <rpc:result>m:status</rpc:result>
            <m:status>confirmed</m:status>
            <m:code>FT35ZBQ</m:code>
            <m:viewAt>
            http://travelcompany.example.org/reservations?code=FT35ZBQ
            </m:viewAt>
        </m:chargeReservationResponse>
    </env:Body>
</env:Envelope>
```

SOAP Web Method Feature

- Allows applications full control over the choice of a “Web method”
  - GET, POST, PUT, DELETE
- Ensure that applications using SOAP can do so in a manner which is compatible with the architectural principles of the World Wide Web
- Supported by the SOAP HTTP binding

SOAP HTTP Binding

- Leverages HTTP request and response model
  - HTTP implicitly correlates its request message with its response message
  - A SOAP application can choose to infer a correlation between a SOAP message sent in the body of a HTTP request message and a SOAP message returned in the HTTP response
- HTTP identifies the server endpoint via a URI
  - URI serves as the identification of a SOAP node at the server

Two Exchange Patterns

- SOAP request-response message exchange pattern
  - Use of the HTTP POST method for conveying SOAP messages in the bodies of HTTP request and response message
- SOAP response message exchange pattern
  - Use of the HTTP GET method in a HTTP request to return a SOAP message in the body of a HTTP response

Which One to Use?

- SOAP request-response message exchange pattern
  - Use it when information resource is manipulated
- SOAP response message exchange pattern
  - Use it when an application is assured that the message exchange is for the purpose of information retrieval
  - Information resource is “untouched” as a result of the interaction
  - Safe and idempotent

SOAP HTTP GET Usage

- Response to a HTTP GET request is a SOAP message in the HTTP response
  - Instead of html or xhtml for browser consumption
  - Data centric
  - Leverages SOAP framework for expressing some application-specific feature through the use of SOAP headers
- HTTP Accept header is used to indicate the preferred representation of the resource being requested
  - "application/soap+xml"

Example : HTTP Get Request

GET /travelcompany.example.org/reservations?
code=FT35ZBQ HTTP/1.1

Host: travelcompany.example.org

Accept: text/html;q=0.5, application/soap+xml

- The HTTP Accept header is used to indicate the preferred representation of the resource being requested

Example: HTTP GET Response

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset="utf-8"
Content-Length: nnnn

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
      <env:Header>
            <m:reservation     xmlns:m="http://travelcompany.example.org/
reservation"
            env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
env:mustUnderstand="true">
<m:reference>uuid:093a2da1-q345-739r-ba5d-pqff98fe8j7d</m:reference>
<m:dateAndTime>2001-11-30T16:25:00.000-05:00</m:dateAndTime>
</m:reservation>
</env:Header>
<env:Body>
...
</env:Body>
</env:Envelope>
```


SOAP HTTP POST Usage

- Used for both exchange of general XML data (Document-driven) or RPC
- HTTP Content-type header must be “application/soap+xml”
- The combination of HTTP Post and Host headers represents URI of the resource
  - http://travelcompany.example.org/Reservations
- If error occurs, HTTP 500 “Internal Server Error” is returned along with an embedded SOAP message containing a SOAP fault

Example: HTTP POST Request

HTTP POST Request
POST /Reservations HTTP/1.1
Host: travelcompany.example.org
Content-Type: application/soap+xml; charset="utf-8"
Content-Length: nnnn

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
<env:Header>
<t:transaction
xmlns:t="http://thirdparty.example.org/transaction"
env:encodingStyle="http://example.com/encoding"
env:mustUnderstand="true" >5</t:transaction>
</env:Header>
<env:Body>
<m:chargeReservation ...
...
</env:Body>
</env:Envelope>
```


Example: HTTP Post Response

HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset="utf-8"
Content-Length: nnnn

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
>
    <env:Header>
        ...
        ...
    </env:Header>
    <env:Body>
        ...
        ...
    </env:Body>
</env:Envelope>
```

Example: HTTP Response with Failure

```
HTTP/1.1 500 Internal Server Error
Content-Type: application/soap+xml; charset="utf-8"
Content-Length: nnnn

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
      <env:Body>
            <env:Fault>
                  <env:Code>
                  <env:Value>env:Sender</env:Value>
                  <env:Subcode><env:Value>rpc:BadArguments</env:Value></env:Subcode>
                  </env:Code>
                  <env:Reason>
                        <env:Text xml:lang="en-US">Processing error</env:Text>
                        <env:Text xml:lang="cs">Chyba zpracování</env:Text>
                  </env:Reason>
                  <env:Detail>
                        <e:myFaultDetails                                    xmlns:e="http://
travelcompany.example.org/faults" >
                        <e:message>Name does not match card number</e:message>
                        <e:errorcode>999</e:errorcode>
                  </e:myFaultDetails>
                  </env:Detail>
                  </env:Fault>
      </env:Body>
</env:Envelope>
```


Example: SMTP Request

```
From: a.oyvind@mycompany.example.com
To: reservations@travelcompany.example.org
Subject: Travel to LA
Date: Thu, 29 Nov 2001 13:20:00 EST
Message-Id:
<EE492E16A090090276D208424960C0C@mycompany.example.com>
Content-Type: application/soap+xml

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
<env:Header>
...
</env:Header>
<env:Body>
...
</env:Body>
</env:Envelope>
```

Example: SMTP Response

From: reservations@travelcompany.example.org
To: a.oyvind@mycompany.example.com
Subject: Which NY airport?
Date: Thu, 29 Nov 2001 13:35:11 EST
Message-Id: <200109251753.NAA10655@travelcompany.example.org>
In-replyto:<
EE492E16A090090276D208424960C0C@mycompany.example.com>
Content-Type: application/soap+xml

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
<env:Header>
...
</env:Header>
<env:Body>
...
</env:Body>
</env:Envelope>
```


Agenda

- What is and What is Not SOAP?
- SOAP Message Structure
- SOAP Terminology
- SOAP Message Exchange
- Document vs. RPC
- SOAP Encoding

RPC vs. Document-style

RPC	Document-Style
Procedure call	Business documents
Method signature	Schema
Tightly-coupled	Loosely coupled
Synchronous	Asynchronous
Typically within intranet	Typically over internet

When to use Which Model?

RPC	Document-Style
Within Enterprise	Between enterprise to enterprise
Simple, point-to-point	Complex, end to end with intermediaries
Shorting running business process	Long running business process
Trusted environment	Blind trust
Reliable and high bandwidth	Unpredictable bandwidth

Document-style Web Services Support

- Use of "document/literal" SOAP message (instead of "RPC/encoding")
  - SOAP body contains XML document, i.e. Purchase order
  - Specified via "style" and "use" attribute in WSDL document
- Use of Attachments
  - Attachment contains XML documents
  - Specified via MIME binding in WSDL documents

Document-Style WSDL

- WSDL provides a document-style service contract between sender and receiver
- Abstract Message Description
  - Provides name for each part
  - Provides type of each message part
- Binding Description
  - Provides messaging packaging format

Document-Style SOAP

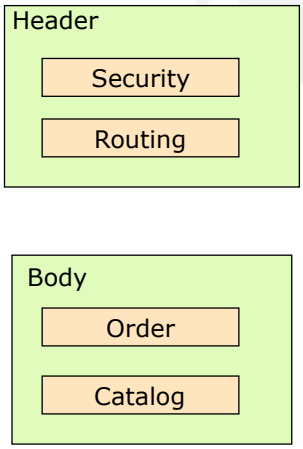
Header

- Specifies message-level services

Payload

- Opaque
- Schema-defined
- Large
- Complex

Envelope



SOAP Security

- Transport level security
  - SOAP over HTTP/S
- SOAP message level security
  - WS-Security
  - ebXML Message Service

Summary

- Web services communicate with their clients via XML-formatted messages
- SOAP (Simple Object Access Protocol) is the messaging protocol of choice for web services
- Some of the advantages of SOAP include
  - More flexible data types
  - Support for headers and extensions
  - Cross-platform support

References

- W3C, “W3C SOAP version 1.2”, http://www.w3.org/TR/soap12-part0/
- Richard Monson-Haefel, “J2EE Web Services”, Addison-Wesley
- Sang Shin, “Web Services Programming Course Root Page”, http://www.javapassion.com/webservices/
- http://www.cs.virginia.edu/~acw/security/doc/Tutorials/SOAP.ppt