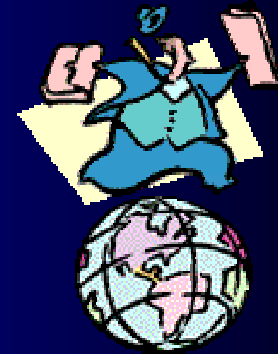


Hitchhiker's Guide to SOAP



Michael Stal, Corporate Technology,
SIEMENS AG Dept. ZT SE 2

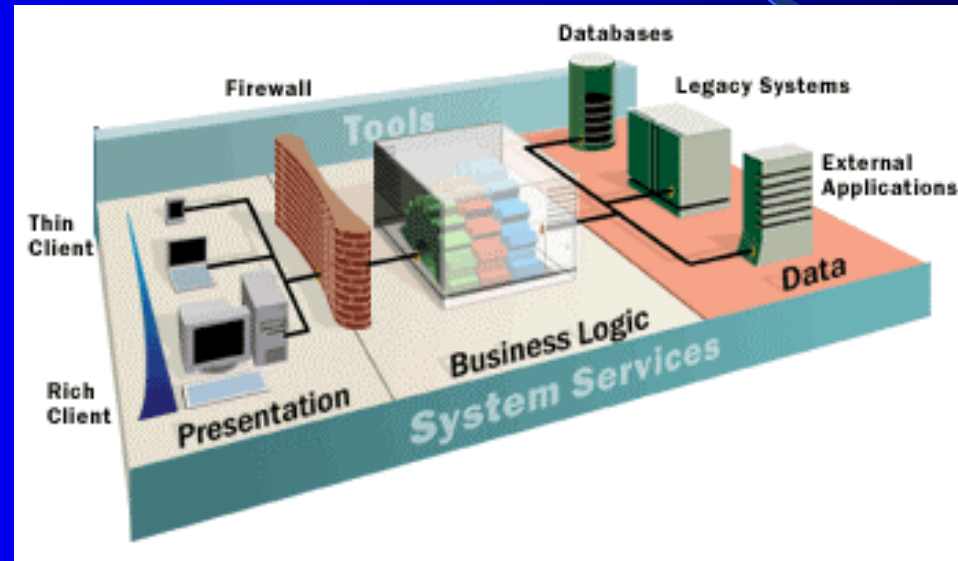
<mailto:Michael.Stal@mchp.siemens.de>

Agenda

- Motivation
- Architectural Foundation
- Web-based RPCs
- SOAP Messages
- SOAP Example
- SOAP Datastructures
- Summary
- References



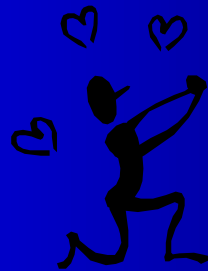
Motivation



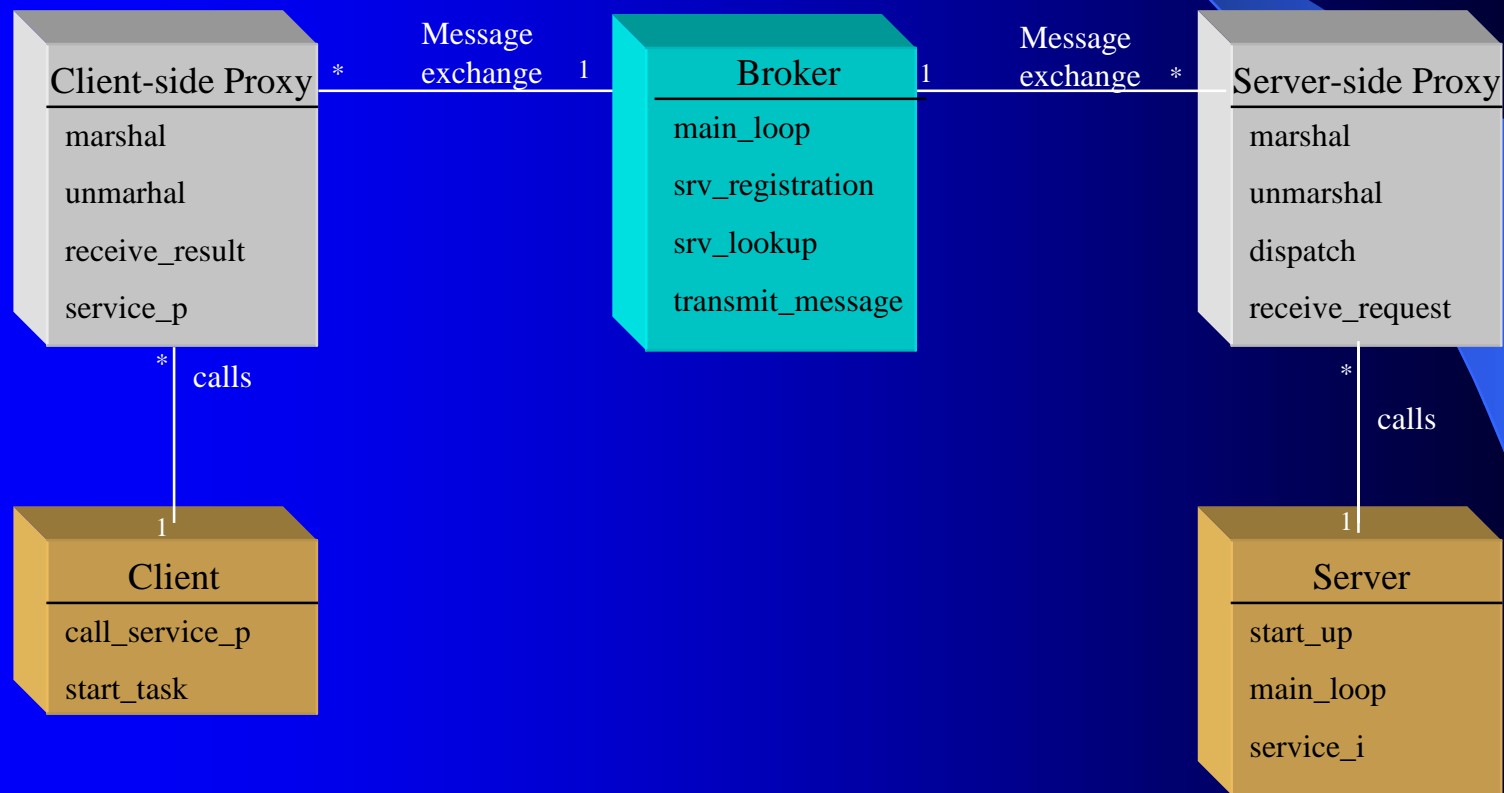
Three-Tier Architectures enable software systems to deploy functionality on the network

Architectural Foundation

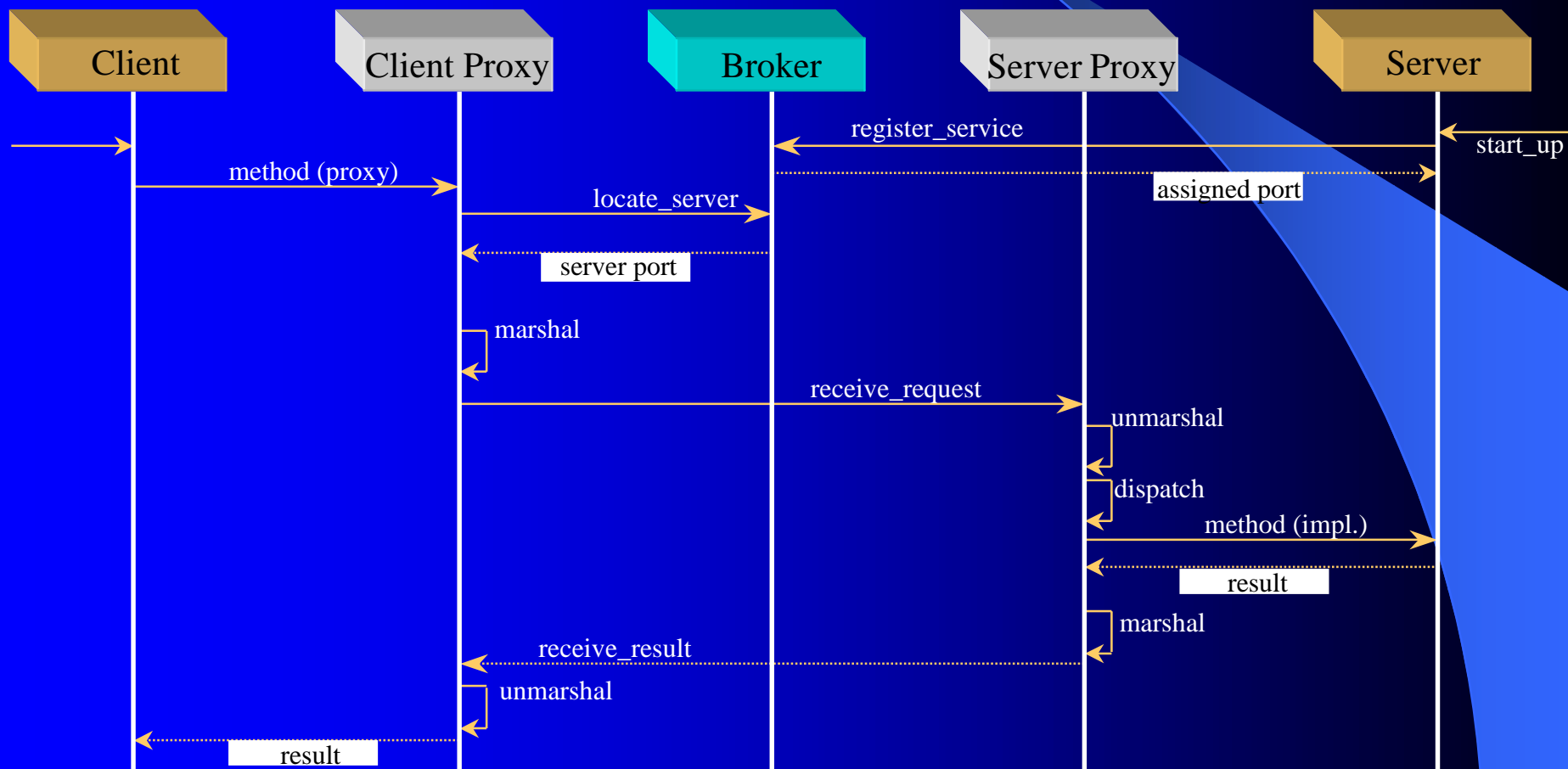
- For this purpose, we need an architecture that
 - supports a remote method invocation paradigm
 - provides location transparency
 - allows to add, exchange, or remove services dynamically
 - hides system details from the developer



Broker Architectures



Typical Scenario



Broker @ Web

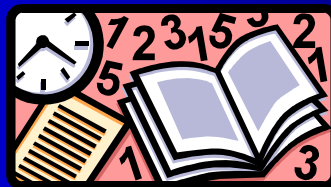
- Firewalls are open for specific ports only such as port 80h.
- However, COM+, CORBA, RMI assign ports dynamically.



Idea: Leverage HTTP as underlying transport protocol

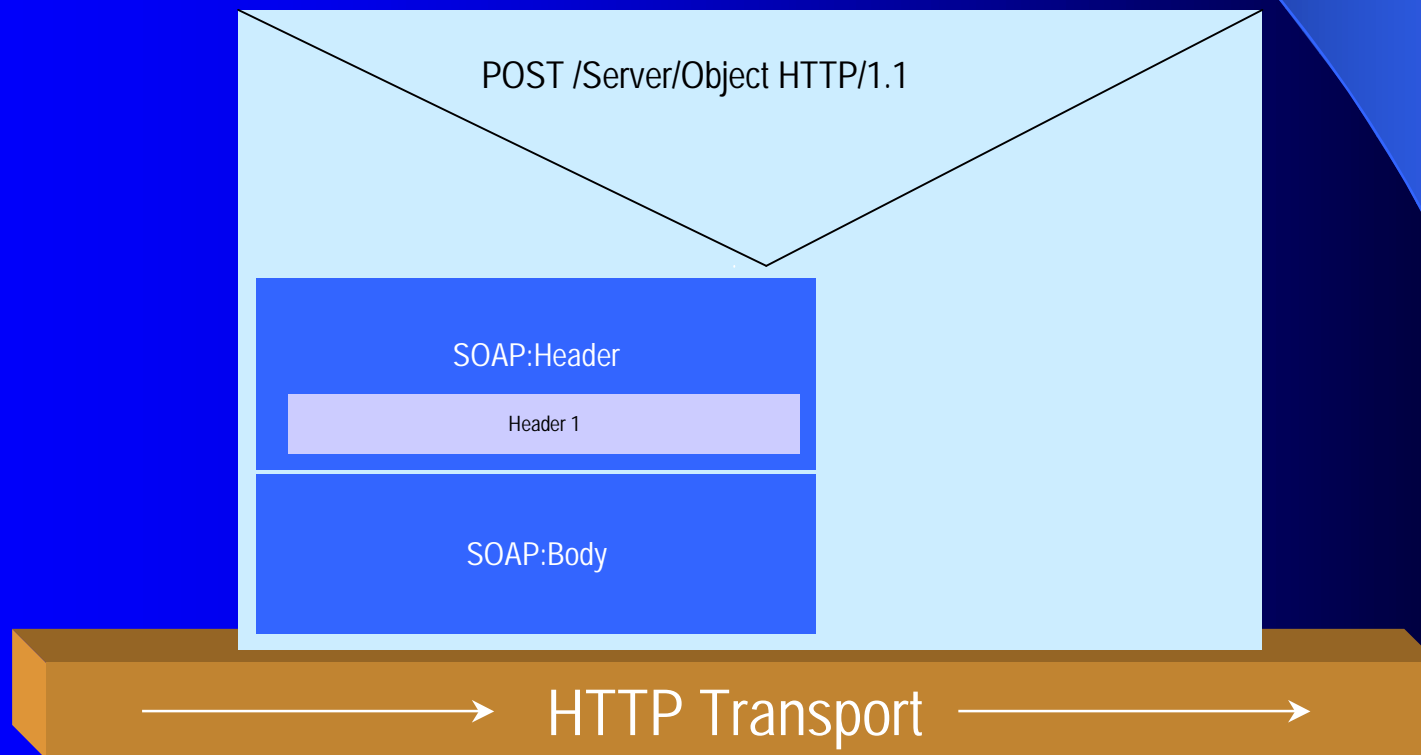
Broker @ Web (cont'd)

- A protocol defines syntax, semantics and order of messages exchanged between peers.
- Hence, we must also introduce a data representation.
- If different brokers interoperate, data must be self-describing.
- Idea: Define data representation with XML



XML + HTTP = SOAP

- The Simple Object Access Protocol defines an XML-based RPC protocol:



SOAP Messages

- SOAP defines two kinds of on-the-wire messages: requests and responses.
- Requests denote the target to be called, as well as the in and in/out-parameters.
- Responses contain an error code or the result, as well as in/out- and out-parameters.
- Messages are sent via M-POST or POST.



General Message Structure

- Here is a fully-featured SOAP Message:

```
<soap:Envelope>
  <soap:Header>
    <transaction>
      <soap:mustUnderstand=„true“ xmlns=„http://tx.com“>
        <id> 12345678 </id>
      </transaction>
    </soap:Header>
    <soap:Body>
      <m:getPhoneNumber>
        <theName> Bill Gates </theName>
      <m:/getPhoneNumber>
    </soap:Body>
  </soap:Envelope>
```

SOAP messages are bracketed by envelopes

headers are optional

this feature must be supported by the receiver

message body

Example: Request

Client

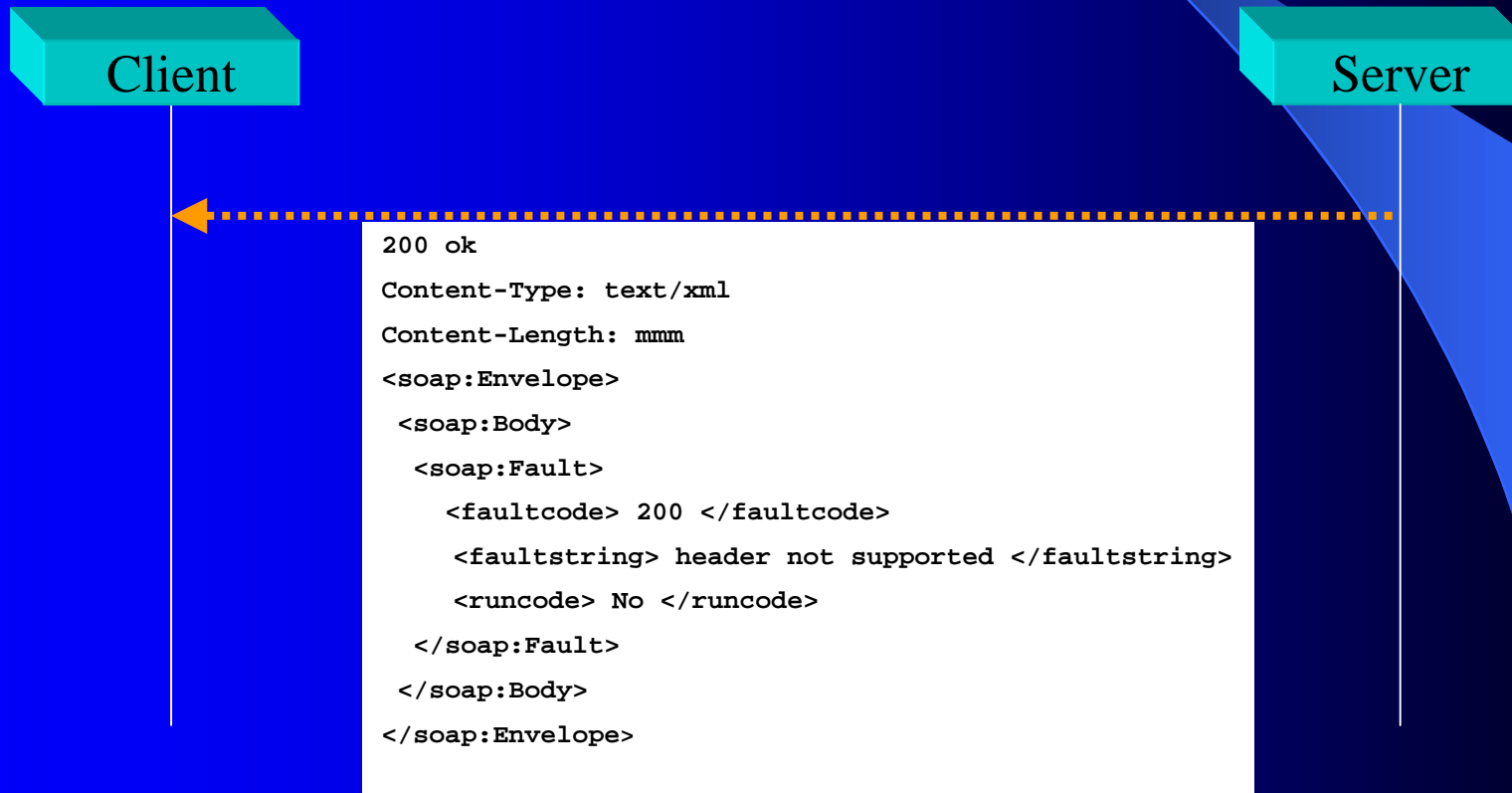
Server

```
POST /magrathea/Shares HTTP/1.1
Host: 139.111.11.13
Content-Type: text/xml
Content-Length: nnn
SOAPMethodName: urn:magrathea.com:Shares:#getValue
<soap:Envelope>
  <soap:Body>
    <m:getValue xmlns:m="urn:magrathea.com:Shares">
      <theShare> MSFT </theShare>
    </m:getValue>
  </soap:Body>
</soap:Envelope>
```

Example: Response (on success)



Example: Response (on application error)



Data Structures

- Record types:

```
<t:Point xmlns:t=„urn:siemens-de:Point“>  
  <x> 12.99 </x>  
  <y> 15.16 </y>  
</t:Point>
```

x, y, color define accessors

- and derived subtypes:

```
<t:ColorPoint xmlns:t=„urn:siemens-de:ColorPoint“>  
  <x> 12.99 </x>  
  <y> 15.16 </y>  
  <color> Red </color>  
</t:ColorPoint>
```

Using Derived Types

- Derived subtypes may be used instead of parent types usingn XSD attribute:

```
<soap:Body>
  <m:drawline>
    <src xsd:type=„t:ColorPoint>
    </src>
      <x> 20.0 </x>
      <y> 25.0 </y>
      <color> Blue </color>
    <dst>
      <x> 10.0 </x>
      <y> 10.0 </y>
    </dst>
  </m:drawline>
</soap:Body>
```

Optionally, „nullable“ parameters can be specified, e.g.,
`<dst xsd:null=„true“/>`

Aliasing

- Referring to values is also supported:

```
<t:Point soap:id=„sharedPoint“>
  <x> 100 </x>
  <y> 200 </y>
</t:Point>
<t:Line>
  <p1>
    <x> 50 </x>
    <y> 60 </y>
  </p1>
  <p2> soap:href=„sharedPoint“ />
</t:Line>
```

Arrays

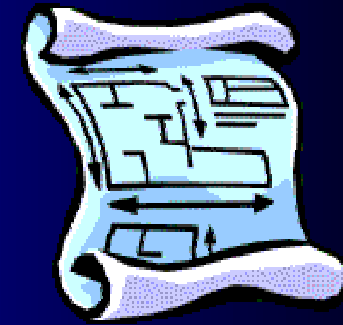
- Arrays and even sparse arrays are available:

```
<t:Rectangle xsd:type="t:Point[4]">  
  <Point> <x> 0 </x> <y> 0 </y> </Point>  
  <Point> <x> 1 </x> <y> 0 </y> </Point>  
  <Point> <x> 1 </x> <y> 1 </y> </Point>  
  <Point> <x> 0 </x> <y> 1 </y> </Point>  
</t:Rectangle>
```

Summary

- SOAP defines a transport protocol based upon XML and HTTP.
- Leveraging HTTP helps clients and servers to circumvent firewalls.
- Using SOAP, web servers can be extended to full blown Broker architectures.
- Microsoft and CORBA vendors have already signaled their support for SOAP.
- Competitors: OMG XIOP, XML RPC

References



- **Don Box:** *A Young Person's Guide to the Simple Object Access Protocol*; MSDN Magazine; March 2000
- SOAP Specification:
<http://msdn.microsoft.com/workshop/xml/general/soapspec-v1.asp>
- **Aaron Skonnard:** *SOAP: The Simple Object Access Protocol*; MSDN Online January 2000;
<http://www.microsoft.com/mind/0100/soap/soap.asp>
- Implementations under <http://www.develop.com>
- **Michael Stal:** *Entfernte Methodenaufrufe a la XML: Prägnate Post*; Heise Verlag; ix Heft 5/2000
- **F. Buschmann, H. Rohner, R. Meunier, P. Sommerlad, M. Stal:** *Pattern-Oriented Software Architecture – A System of Patterns*, Wiley, 1996. New volume on networked objects under construction.