Network Services, VU 2.0

Middleware Protocols

(WebServices, IIOP, RMI, .NET Remoting)

Dipl.-Ing. Johann Oberleiter
Institute for Informationsystems, Distributed Systems Group

Agenda

- Web services / SOAP
- RMI
- IIOP

Web services

- UDDI consortium
  - "self-contained, modular business application that have open, Internet-oriented, standards-based interfaces"
- W3C
  - "a software application identified by a URI, whose interfaces and bindings are capable of being defined, described, and discovered as XML artifacts. A Web service supports direct interactions with other software agents using XML-based messages exchanged via Internet-based protocols

Simple Object Access Protocol

- Minimal possible infrastructure necessary to perform RPC through Internet
  - Use of XML for message header & body
  - Internet protocols (HTTP/SMTP) used for transportation
- SOAP consists of
  - Envelope construct: overall structure of message
  - Encoding rules: serialization of application data types
  - SOAP RPC: representation of remote procedure calls
  - Binding framework – to protocols (HTTP,SMTP,...)
  - Fault handling
- Advanced message processing
  - Forwarding intermediaries – route messages based on the semantics of message
  - Active intermediaries – process/modify messages before forwarding

SOAP / 2 - Messages

- SOAP messages
  - Envelope: top element of XML element
  - Header
    - Elements are application-specific
    - May contain context information (eg. transaction contexts)
    - May be changed by intermediaries
  - Body
    - Elements are application-specific
    - Applications put their data there
    - Processed by recipient only

SOAP / 3 - Example

```
<?xml version="1.0">
<env:Envelope
   xmlns:env="http://www.w3.org/2002/12/soap-envelope">
```

```
<env:Header>
  <m:reservation xmlns:m=http://travelcompany.example.org/reservation
   env:role=http://.../role/next
   env:mustUnderstand="true"
   <m:dateAndTime>2005-05-29 20:00:00</m:dateAndTime>
</env:Header>
```

```
<env:Body xmlns:p=...>
  <p:departure>Vienna</p:departure>
  <p:arriving>Frankfurt</p:departure>
  <p:departureDate>2005-06-20 14:00:00</p:departureDate>
</env:Body
```

```
</end:Envelope>
```

SOAP / 4

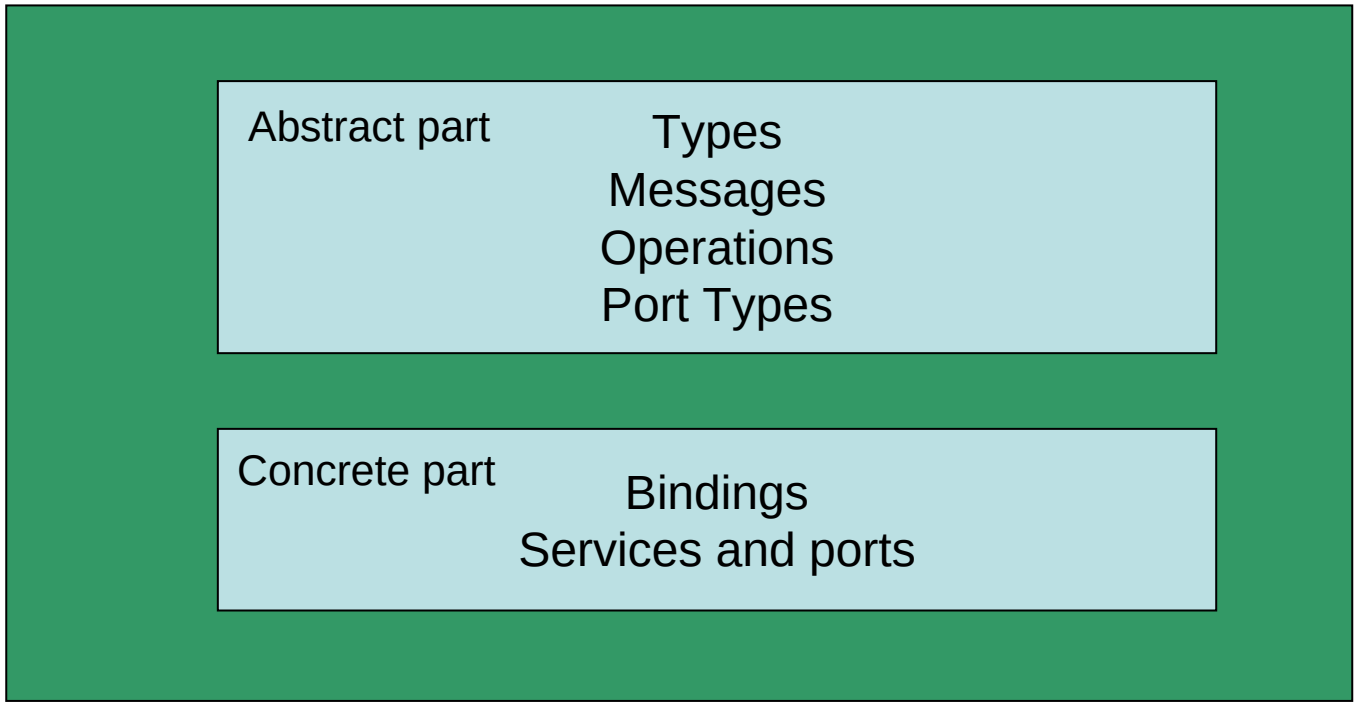
- RPC – style operation
  - SOAP message encapsulates the request
  - Another SOAP message encapsulates the response
  - Body contains actual call
    - Including name of procedure and input parameters
- Document – style operation
  - Interacting applications agree upon structure of documents
  - Body contains not necessarily the name of a procedure

SOAP / 5

- Processing nodes may play 1+ Roles
  - Block in SOAP header may include intended role name
  - Predefined role names
    - "none" – should not be processed, may be read
    - "next" – every node may process this node
    - "ultimateReceiver – should only be processed by recipient
  - "mustUnderstand" flag
    - Block must be processed

Web service Description Language WSDL

- WSDL specification
  - XML file



WSDL – Elements / 1

- Types
  - Default WSDL uses XML Schema
    - Different type system may be specified
  - Define all data structures that will be exchanged with messages
- Messages
  - In WSDL a messages is a typed document divided into parts
    - Part characterized by name and by type
    - Resembles parameters in method invocations
- Operations
  - Defines operations and which messages they use for input/output
  - Transmission primitive
    - One-way – endpoint receives a message
    - Request-response – endpoint receives a message, sends a correlated message
    - Solicit-response – endpoint sends a message, receives a correlated message
    - Notification - endpoint sends a message
  - Which transmission primitive an operation follows is determined by order and availability of input/output message
- Port Types
  - Grouping of operations
  - In WSDL 1.2 port types may extend port types

WSDL – Elements / 2

- Interface bindings
  - Message encoding for port type
  - Protocol binding for port type
  - Encoding rules for serializing parts of messages into XML
    - "Literal" encoding uses WSDL types defined in XML Schema, literally uses those definitions
      - Primarily used for Document-style interaction
    - "SOAP" encoding transfers WSDL types into XML using SOAP encoding rules
      - Primarily used for RPC-style interaction
- Ports
  - Defines Endpoints
  - Combines Interface binding information with URIs
- Service
  - Logical grouping of ports

Universal Description, Discovery, and Integration

- UDDI standard for describing, publishing, and finding Web services
  - Evolving
  - Can be accessed itself via Web services
- White pages
  - Listings of organisations
- Yellow pages
  - Classification of organization based on categories
- Green pages
  - Technical description of services offered by registered organizations
  - How a given Web service can be invoked

UDDI / 2

- Main entities
  - "businessEntity"
    - describes organization that provides a Web service
  - "businessService"
    - describes a group of related Web services offered by a businessEntity
  - "bindingTemplate"
    - Technical information necessary to use a particular Web service
    - Address of the Web service
  - "tModel"
    - "technical Model"
    - Generic container for any kind of specification
    - Eg. WSDL interface, interaction protocol, semantics of the operation

RMI Protocol

- Remote Method Invocation
  - Java RPC
  - See Distributed Systems Lab
- RMI Protocol
  - Stream based
    - In & Out streams of corresponding socket pair
      - As consequence only header information required on input stream is acknowledgement (0x4e)
      - Other header information implied by context of stream pairing
  - Uses Object Serialization protocol for Marshaling
    - Representation of Java objects
      - In "Call" and "ReturnValue" messages

Output Stream

- Transport Header
  - 0x4a 0x52 0x4d 0x49 (JRMI)
  - Version (0x00 0x01)
  - Protocol
    - StreamProtocol (0x4b)
    - SingleOpProtocol (0x4c)
      - Used for interactions embedded in HTTP requests
    - MultiplexProtocol (0x4d)
- Messages (one or more)
  - Call
    - Method invocation
    - Contains "0x50 CallData"
  - Ping
    - Testing liveness of remote VM
    - Contains "x52´"
  - DgcAck
    - Acknowledgement directed to server's distributed Garbage Collector
    - Contains "0x54 UniqueIdentifier"

Input Stream

- Protocol Acknowledge
  - 0x4e
  - 0x4f in case protocol not supported
- Returns (one or more)
  - ReturnData
    - "0x51 ReturnValue2
  - PingAck
    - "0x53"

RMI & HTTP Post

- Invocation of RMI through firewall
  - Use of HTTP POST
    - "http://<host>:<port>/
      - Direct communication with RMI server on host and port
    - "http://<host>:80/cgi-bin/java-rmi?forward=<port>
      - Invokes CGI script on the server which forwards invocation to server on specified port
  - Automatically used by RMI
    - Client first attempts direct connection without HTTP
    - In exception case tries HTTP connection
      - If java.net.noRouteToHostConnection or java.net.UnknownHostException thrown
    - Server socket automatically detects if it was a HTTP POST request

RMI Multiplexing

- Only one endpoint is able to open bidirectional connection
  - Eg. Security managers (eg. applets) may disallow server sockets
    - Instead may open normal socket connection
- Allows multiple virtual connections exist in parallel
- Operations
  - OPEN, CLOSE, CLOSEACK, REQUEST, TRANSMIT

Internet Inter-ORB Protcol (IIOP)

- Transport protocol of CORBA
  - Also useable in RMI
  - Alse useable in EJB
- Special form of GIOP (General Inter-ORB Protocol)
  - IIOP uses TCP
- Communication between Object Request Broker
  - ORB responsible for
    - Find object implementation
    - Receive & invoke request on objects
- Few, simple messages
- Very efficient

IIOP / 2

- Transmission data represented in CDR
  - Common Data Representation
  - How CORBA IDL Data Types are represented in transmission packets
  - Sender defines byte ordering (Little/Big endian)
  - Primitive Types aligned on natural boundaries

IIOP Messages

- Request (0, originates at Client)
  - Operation request
- Reply (1, Server)
  - Reply of operation
- CancelRequest (2, Client)
  - Signals client is no longer interested in result
- LocateRequest (3, Client), LocateReply (4, Server)
  - Determines if server is capable of receiving requests for an object
- CloseConnection (5, Both)
  - One side closes the connection
- MessageError (6, Client+Server)
  - Response to message with invalid Header
- Fragement (7, Client+Server)
  - Message is continuation of previous message
  - Used for large requests

IIOP Request

- GIOP message header
  - char[4] Magic ("GIOP")
  - Version GIOP_Version (eg. 1.2)
  - byte Flags (byte ordering, fragments follow)
  - byte Message Type (0 Request, 1 Reply, …)
  - unsigned long Message Size
- RequestHeader
  - unsigned long Request-id
  - byte response_flags
  - byte reserved[3]
  - TargetAddress target
  - string operation (eg. "sum")
  - ServiceContextList service_context
- Request body
  - parameters marshaled in CDR from leftmost parameter to rightmost
    - Eg. void sum(double x, double y);

IIOP Reply

- GIOP Header
- Reply Header
  - unsigned long request_id
  - ReplyStatusType reply_status
    - Success, Exception
  - ServiceContextList service_context
- Reply Body

.NET Remoting

- "RMI" for .NET
- Possible to configure used channel (=protocol)
  - TCP and HTTP included
- Possible to configure data representation
  - SOAP and binary representation included
- Easy to extend with other protocols
  - Example: J.Oberleitner & T. Geschwind: "Transparent Integration of CORBA and the .NET Framework", in "On the Move to Meaningful Internet Systems, 2003 CoopIS, DOA, and ODBASE
    - Describes how an IIOP Channel can be used with .NET Remoting
    - http://www.infosys.tuwien.ac.at/reports/repository/TUV-1841-2003-20.ps