Java and C# in depth

Carlo A. Furia, Bertrand Meyer

Distribution and web services

From concurrent to distributed systems

	Multiprocessor	Multicomputer	Distributed system
Node configuration	CPU	CPU, RAM, net interface	Complete computer
Node peripherals	All shared	Shared excluding maybe disks	Full set per node
Location	Same rack	Same room	Possibly worldwide
Internode communication	Shared RAM	Dedicated interconnect	Traditional network
Operating systems	One, shared	Multiple, same	Possibly all different
File systems	One, shared	One, shared	Each node has own
Administration	One organization	One organization	Many organizations

From: A. S. Tanenbaum, *Modern operating systems*, 3rd edition, 2009.

Models of distributed systems

- There are many different models of distributed computing
  - Document-based (e.g., the WWW)
  - File-system based (e.g., NFS, Samba)
  - Object-oriented middleware (e.g., CORBA)
  - Tuple spaces (e.g., Linda)
  - Publish/subscribe (e.g., IBM Websphere MQ)
  - Grids
  - ...
- In this class we' re presenting the web service model

Outline

- What’ s a web service
- Protocols for web services
  - WSDL
  - UDDI
  - SOAP
- Web services styles
  - RPC
  - SOA
  - RESTful
- Web services development styles
- Assessment of web services

Java and C# in depth

Carlo A. Furia, Bertrand Meyer

What’s a web service

What’ s a web service?

A (relatively) recent technology for distributed computing

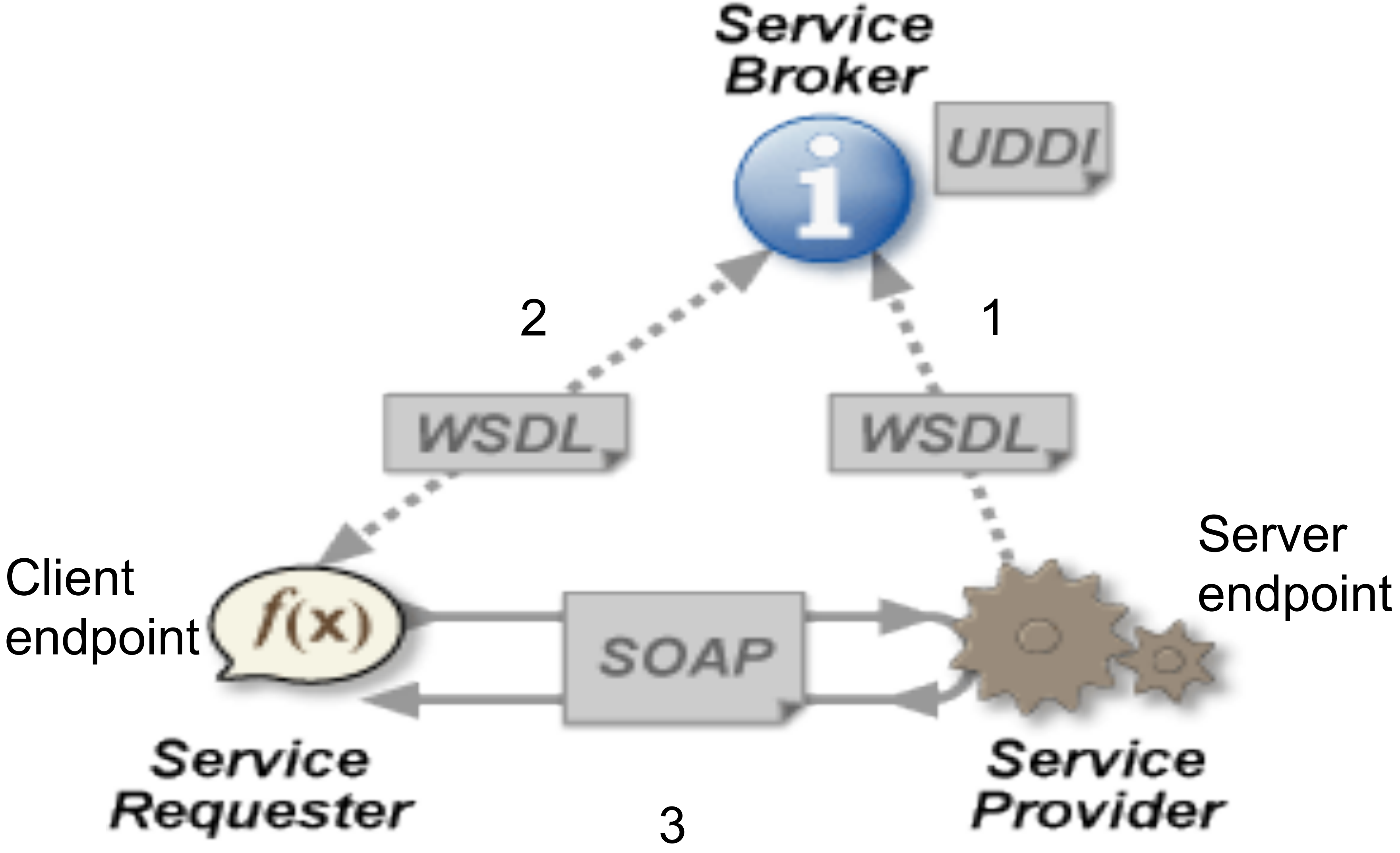
- *A software system designed to support interoperable machine to machine interaction over a network* -- W3C
- A self-contained, self-describing client-server system that enforces communication via XML messages
- A web API accessible over a network, executed on a remote system hosting the requested service
- A buzzword (http://en.wikipedia.org/wiki/List_of_buzzwords)

Web services and protocols

Web services (WSs) combine a variety of protocols to define, locate, implement, and make functionalities interact

- Description protocol layer: WSDL
    - define interface and configuration
- Discovery protocol layer: UDDI
    - registry to locate WSs
- Messaging protocol layer: SOAP (XML), RESTful
    - communication (higher-level than transport)
- Transport protocol layer: HTTP(S), SMTP, FTP, RSS, XMPP

A high-level view



Source: http://en.wikipedia.org/wiki/Web_services

Java and C# in depth

Carlo A. Furia, Bertrand Meyer

Protocols for web services

The WSDL protocol

Web Services Description Language

- pronounced "wiz-dal" or "W-S-D-L"
- XML-based, platform independent
- Version 2.0 is endorsed by W3C
- Integrated in Microsoft's .NET platform

Describes:

- The public interface of a web service
- Details of protocol bindings
- Configuration data

Used to generate client and server code stubs from an abstract description

The WSDL protocol

WSDL describes WSs as collections of network endpoints (a.k.a. ports) via an XML document

- A port type defines an abstract collection of supported operations
  - it is an abstraction of the concrete port

- A (concrete) port is defined by associating a network address to a reusable binding

- A collection of ports defines a service

The WSDL protocol

- A reusable binding is a concrete protocol and data format specification for a specific port type
  - similar to a mapped operation

- Through the binding, operations and messages are bound to a concrete network protocol and message format

- A message is an abstract description of the exchanged data

Abstraction: the abstract definition of ports and messages and their implementations are uncoupled, thus allowing reuse

The UDDI protocol

Universal Description, Discovery and Integration

- pronounced “Yu dee”
- XML-based, platform independent

A public registry for businesses on the WWW

- Each business publishes a list of services

Each UDDI business registration consists of:

- White pages: address, contact and known identifiers
- Yellow pages: industrial categorizations
- Green pages: technical info about offered services

A brief history of UDDI

- Written in the year 2000

- Enforces a vision in which consumers link to providers through a public brokerage system

- By the end of 2005, 70% of Fortune-500 companies planned to use it

- In January 2006, IBM, Microsoft, and SAP announced they were discontinuing their public UDDI nodes
  - anyway, according to Microsoft and IBM, the interoperability and robustness of the UDDI was proven

A brief history of UDDI

The idea was that companies could publish how they wanted to interact, and other companies could find that information and use it to establish a relationship. Needless to say, this isn't how companies do business. There's always a human element to establishing a relationship.

-- Jason Bloomberg, senior analyst at ZapThink

The UDDI protocol today

- Mostly used in intranets (within-the-firewall)

- Used as metadata management standard for Service Oriented Architectures (more on this later)

The SOAP protocol

Service Oriented Architecture Protocol
(formerly Simple Object Access Protocol)

- Simple and extensible XML-based communication protocol
- Platform and language independent
- Is a basic message-wrapping framework
- Has bindings to lower-level protocols such as HTTP, HTTPS, SMTP, FTP, RSS, XMPP, ...

Java and C# in depth

Carlo A. Furia, Bertrand Meyer

Styles in web service design

Styles of use of WSs

- A WS combines multiple communication protocols
- Different “styles” combine the protocols to build a distributed application
  - Remote Procedure Call WSs (RPC)
  - Service Oriented Architecture of WS (SOA)
  - Representational Transfer State (REST)
- The classification is not rigid at all, and hybrid solutions are common

RPC: Remote Procedure Call WSs

The RPC style uses WSs technology to implement a client-server model of distributed computing

- Distributed method calls available to clients
  - method invocation is location-transparent
- The basic unit of service is usually the WSDL operation
  - operation-oriented style
- How an RPC works:
  - a client (a network node) initiates the call by sending a request to a server (also a network node)
  - the server responds immediately, while the client blocks
  - the call can fail due to unpredictable network issues

RPC: Remote Procedure Call WSs

The RPC style uses WSs technology to implement a client-server model of distributed computing

- Middleware technologies with distributed objects offer variants of RPC
  - Java RMI (Remote Method Invocation)
  - Microsoft's DCOM (Distributed Component Object Model)
  - OMG's CORBA (COmmon Request Broker Architecture)
  - (they all pre-date WS technologies)

- In RPC WS there is not much decoupling between offered services and local implementations

- Today, RPC-style WSs are mostly discontinued

SOA: Service-Oriented Architecture WSs

The SOA style builds a message-oriented distributed application

- The message is the basic unit of communication
  - message-oriented style
- Typically uses SOAP for communication
  - may use RPC but only as implementation medium
- In SOA WSs there is a loose coupling between offered services and local implementations
- The focus is on the WSDL contract, specifying:
  - Header (name, version, owner, type, …)
  - Functional aspects (functionality accomplished, service operations, invocation details, ...)
  - Non-functional aspects (security constraints, allowed failure rate, service level agreement, ...)

RESTful WSs: Representational Transfer State

The RESTful style provides an interface with well-known, standard operations to interact with stateful resources

- Application state and functionality are divided into resources
- Every resource is uniquely addressable using a universal syntax (usable in hyperlinks)
- All resources share a uniform interface for the transfer of state between client and resource
  - A constrained set of well-defined operations (e.g., GET, POST, PUT, DELETE for HTTP)
  - A constrained set of content types
- Protocols are client-server, stateless, cacheable, and layered

- Q) What's a large distributed application designed in the RESTful style?

RESTful WSs: Representational Transfer State

The RESTful style provides an interface with well-known, standard operations to interact with stateful resources

- Application state and functionality are divided into resources
- Every resource is uniquely addressable using a universal syntax (usable in hyperlinks)
- All resources share a uniform interface for the transfer of state between client and resource
  - A constrained set of well-defined operations (e.g., GET, POST, PUT, DELETE for HTTP)
  - A constrained set of content types
- Protocols are client-server, stateless, cacheable, and layered

- Q) What’s a large distributed application designed in the RESTful style?
- A) The WWW, using URIs, MIME types, HTTP, HTML, DNS, ...

WS styles comparison: RESTful vs. RPC

- RESTful focuses on resources
  - Resources are standardized
  - Commands are defined by simple combinations of resources that are retrieved, stored, set, ...
  - Depends on functionalities of the network infrastructure (e.g., caching and authentication for HTTP)

- RPC focuses on commands/operations
  - Commands are customized
  - Commands are defined by custom methods of varying complexity (depending on practices)
  - Increase the coupling between service provider (server) and consumer (client)

Java and C# in depth

Carlo A. Furia, Bertrand Meyer

Styles in web service development

Contract-first and contract-last WSs

Contract-first:

1. Write WSDL contract
2. Generate a skeleton implementation in the language of choice
3. Complete the implementation of the service that meets the contract

Contract-last:

1. Write an implementation of the service in the language of choice
2. Abstract the behavior of the implementation at the interface as a WSDL contract

Issues with contract-last development

- Object/XML impedance mismatch
  - XML used for WSDL contracts
  - Object-oriented (OO) language used (typically) for implementation
- Fragility and performance
- Reusability and versioning

Object-XML impedance mismatch

Mismatch between the XML and OO models

  - XML and XSD are hierarchical data languages
    - They essentially describes trees
    - XSD (a.k.a. XML Schema) defines templates for (classes of) XML documents
  - The implementation language is (typically) OO
- XSD's notion of "extension" is not conformant to OO inheritance
  - E.g., no overriding possible, attributes are not inherited, ...
- Effects of the mismatch:
  - Non-portable types

    (from implementation language to XML)
  - Cyclic references (A → B → A)

Fragility and performance

- Different SOAP stack implementations may generate different web service contracts from an implemented class/application

- Automatic translation from an (OO) implementation language into XML may leak complex information to the network
  - Dependencies in the implementation can be deep, entangled, and difficult to foresee

- This does not happen in contract-first development, because a contract is simpler to understand fully and guarantees a certain level of encapsulation

Reusability and versioning

Maintaining a separate XSD schema definition for contracts allows reuse in different scenarios.

- In contract-last development, any contract change is first reflected in a new class/interface definition. The previous version of the code must still be mantained for clients that still rely on it.

- In contract first development, the old and new versions of the contract can be implemented in the same class (possibly using conversion tools for XML such as those based on XSLT)

Java and C# in depth

Carlo A. Furia, Bertrand Meyer

Assessment of web services

What good are web services?

- Provide interoperability between different web applications running on different platforms

- Represent one popular option out of many possible models of distributed systems

- Focus on a specific tradeoff: interoperability and extensibility over ease-of-use and performance

- Within a single-language environment, you’re probably better off with solutions targeted to that language
  - Java’s RMI, Enterprise Java Beans, Java Spaces, ...
  - .NET framework and components
  - ...