

CS603: Distributed Systems

Lecture 3: Web Services

What are Web Services?

- Software components that allows applications (**different programming languages and different platforms**) to exchange data over computer networks
- Communication is via **SOAP** (uses HTTP)
- Web services can be described in a standard way **WSDL** (uses XML) language

Benefits

- Software components that allows applications (**different programming languages and different platforms**) to exchange data over computer networks



Portability, vendor, platform independence

Benefits

- Communication is via **SOAP** (uses HTTP)



Use of HTTP ensures that web services can work through many common firewall security measures without requiring changes to their filtering rules.

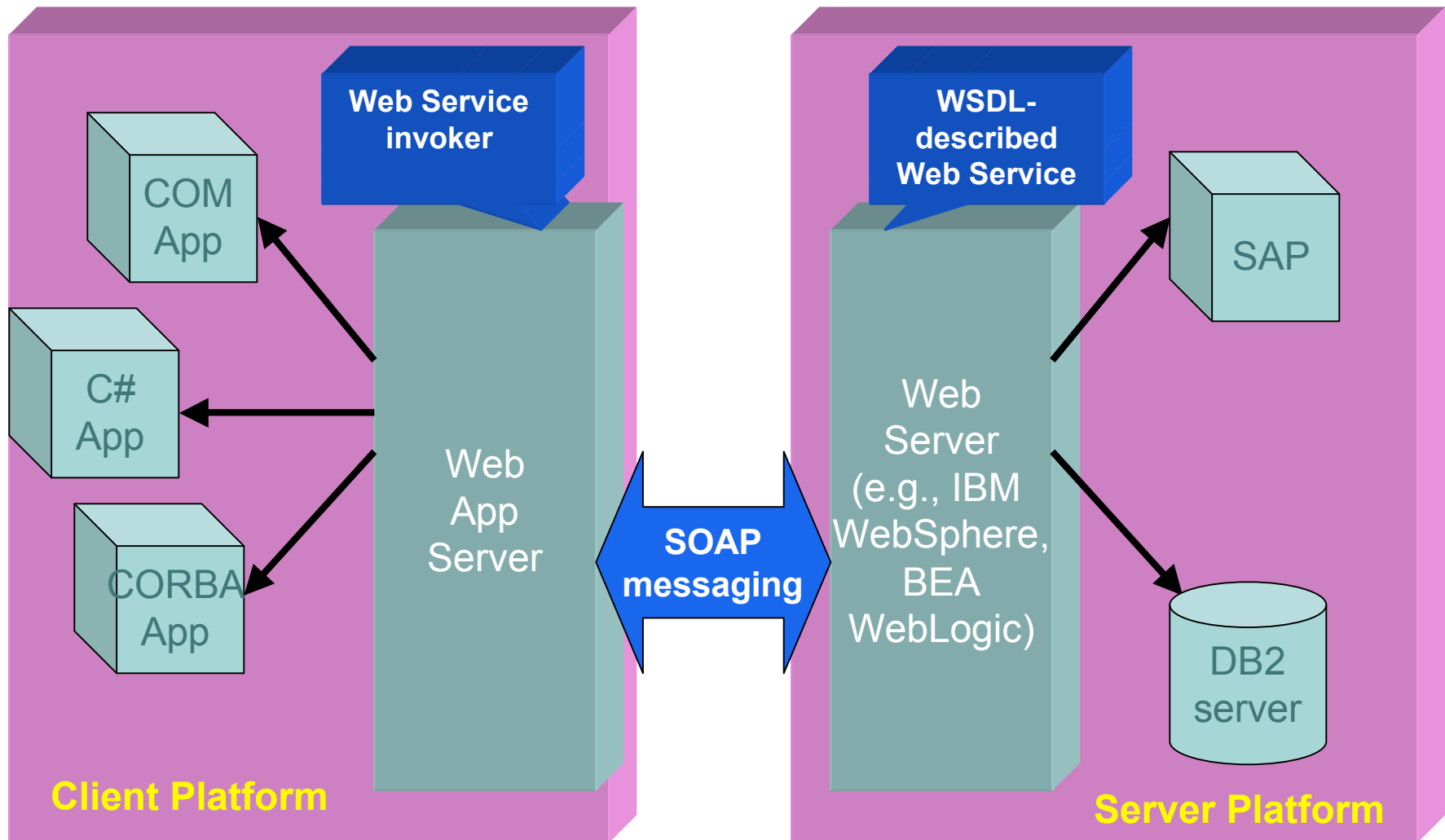
Benefits

- Web services can be described in a standard way **WSDL (uses XML)** language

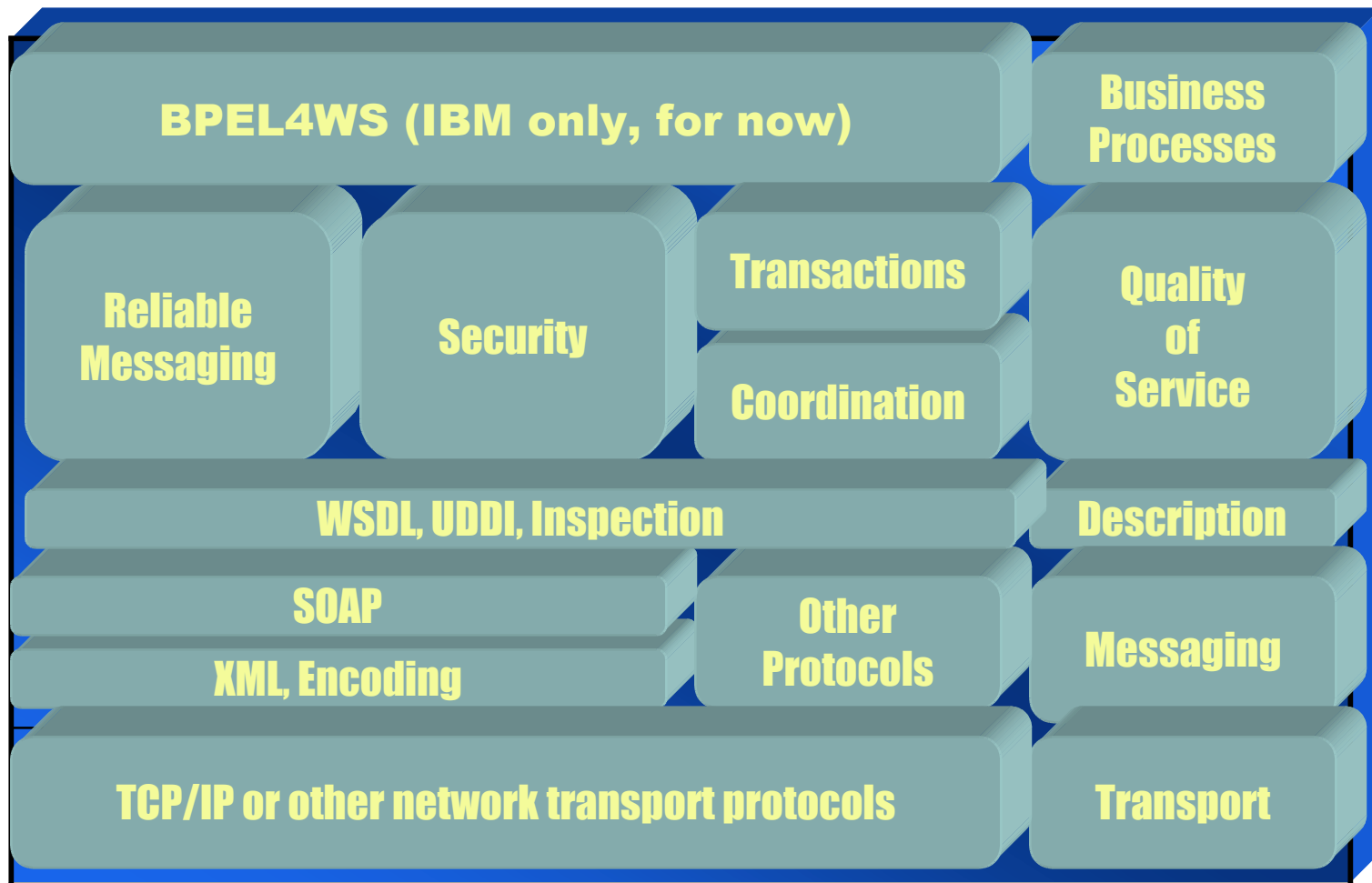


Uses existent open standards and protocols.
Text description make it easy to understand

Web Services Architecture



Web Services Stack



Picture courtesy of <http://www.cs.cornell.edu/ken/book/>

Web Services: Details

- Service must be published
- Client must
 - Discover the service
 - Bind to the server
 - Pack the request (marshaling) and send the SOAP request
- Server must
 - Unpack the request (demarshaling), handles it, computes result.
 - Sends answer back in the reverse direction: from the server to the SOAP router back to the client.
- Communication goes through the SOAP router

SOAP

- a cleansing agent made from the salts of vegetable or animal fats

OOOPs! Wrong definition :)



- **Simple Object Access Protocol SOAP** : lightweight XML-based protocol for exchange of information in a decentralized, distributed environment:
 - an envelope that defines a framework for describing what is in a message and how to process it
 - a set of encoding rules for expressing instances of application-defined datatypes
 - a convention for representing remote procedure calls and responses

WSDL

- Web Service Definition Language
- Uses XML to describe network services as collections of communication endpoints capable of exchanging messages.
- The abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings:
- Allows the reuse of abstract definitions:
 - **messages**: abstract descriptions of the data being exchanged
 - **port types**: abstract collections of operations.

XML

- XML (Extensible Markup Language) allows information and services to be encoded with **meaningful structure and semantics** that computers and humans can understand.
- XML focus on what data is (while HTML focus on how data looks)
- XML document **defines how information is organized**, and NOTHING ELSE

```
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```
- Some other code will do the sending/receiving and displaying.

UDDI

- UDDI (Universal Description, Discovery and Integration): standard for storing naming information about a service
- Application stores a UDDI record in some naming service
- UDDI can be saved into a XML database and queried
- Web Services use UDDI to have unique names for services and facilitate service lookup

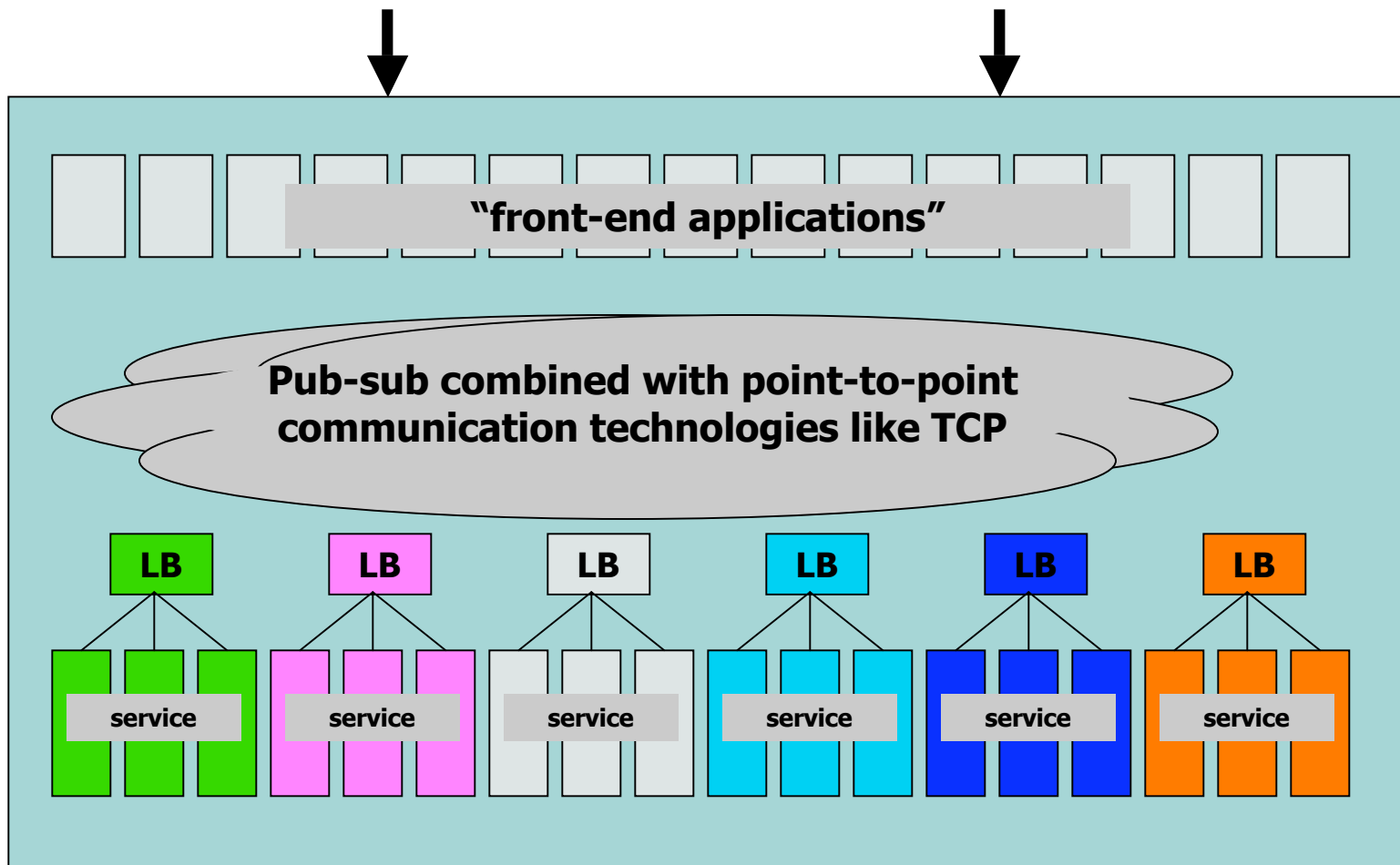
Web Services: Document-Centric

- Web Services relies on documents: all functionality is specified ultimately through documents
 - Client presents the server with a document
 - The server accepts this document and agrees to “route” it for processing
- Corba is ... object-centric

Service Discovery

- Application produces a description using the UDDI standard
- Repository maintains one or more databases
- Client can browse for a good match
- Can this be automated?

The Client Does not Actually Talks with a Server, but a Data Center



Picture courtesy of <http://www.cs.cornell.edu/ken/book/>

Issues with Service Discovery

- Clients do not talk with a server actually, but with a data centers
- Can a data center influence client's decision?
- Since services are clustered, how can a client ask for the 'right' server
- How should caching be handled?
Servers cache data for clients, make sure the client gets the right cache

Where Web Services Fail Short

- Allow the data center to control decisions the client makes
- Provide assistance in implementing naming and discovery in scalable cluster-style services
 - How to load balance? How to replicate data? What precisely happens if a node crashes or one is launched while the service is up?
 - Help with dynamics. For example, best server for a given client can be a function of load but also affinity, recent tasks, etc

The Way It Works Now

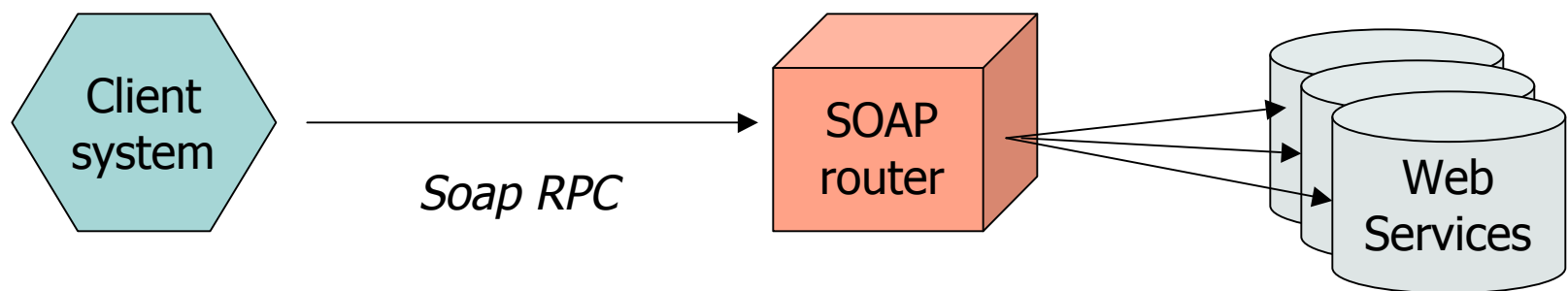
- Client queries directory to find the service
- Server has several options:
 - Web pages with dynamically created URLs
 - Server can point to different places, by changing host names
 - Content hosting companies remap URLs on the fly.
E.g. <http://www.akamai.com/www.cs.cornell.edu>
(reroutes requests for www.cs.cornell.edu to Akamai)
 - Server can control mapping from host to IP addr.
 - Must use short-lived DNS records; overheads are very high!
 - Can also intercept incoming requests and redirect on the fly

Reliability and Transactions Support

- **WS_RELIABILITY:**
 - One operation
 - Target are applications that can tolerate delays in answers from servers, recovering of the server in case of failure does not need to happen immediately
- **WS_TRANSACTIONS:**
 - Set of operations, "all or none" semantic
 - Service that allows the client to interact with the server and obtain ACID (Atomic, Consistent, Isolated, and Durable) guarantees

WS_RELIABILITY

- Client talks with the server through an intermediary (let's call it mailbox)
- **ASSUMPTION: MAILBOX** (queuing system) **NEVER FAILS**
- **Server can fail**



WS_RELIABILITY: How it works

- Client sends request (with unique identifier) to queuing system
- Client specified what semantics he wants in case of failure:
 - Exactly once
 - At most once
 - At least once
- Queuing system logs the request and sends an acknowledgment
- Client moves on and checks from time to time to see what happened with the request

WS_RELIABILITY

- Queuing system
 - Stores all the messages of the client
 - Forwards all the messages it save during a server crash
- DOES NOT WORK if queuing system crashes
- DOES NOT WORK for applications that require immediate answer from the server
- It works for applications that can send a request and can check back later to see if anything happened

WS_TRANSACTIONS

- Requires numerous protocol steps, and writing on the disk
- Basic transactions -- short-running operation sequences on one or more transactional backend services
- Business transactions -- script of basic transactions that includes exception handling logic

How Web Services Deal with Failures

- Failures of
 - naming service
 - backend servers
 - clients
- As other technologies, Web services suffers from:
 - can not distinguish between crash failures and transient failures (crash vs. latency)
 - when the service reports an error client does not know details about what happened

Web Service Applications

- **Grid Computing: Distributed computing**
- Involves coordinating and sharing computing, application, data, storage, or network resources across dynamic and geographically dispersed organizations.
- Research challenges: scalability, security, system management
- Computing power and storage increase, network remains the bottleneck

Future? Autonomic Computing

- Targets large-scale computer systems
- Computers must learn to manage themselves, in accordance with high-level guidance from humans.
- Self-monitoring, self-configuration, self-optimization, self-healing, and/or self-protection.
- Specific self-managing components, such as server, client, database, storage, or network elements. Emphasis should be placed on interactions with other components, or techniques or lessons that may generalize to other components.
- <http://www.autonomic-conference.org/>

Semantic Web

- Describes relationships between types of information
- Data on the web defined and linked in a way that allows automation, integration and reuse of data across various applications.

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation." -- Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001

Next ...

- We will look at fundamental concepts in distributed systems:
 - Time
 - Consistency
 - Detecting failures
 - Membership