

中国科学院深圳先进技术研究院 SHENZHEN INSTITUTE OF ADVANCED TECHNOLOGY CHINESE ACADEMY OF SCIENCES

DISTRIBUTED SYSTEMS

Most concepts are drawn from Chapter 9

Slides are revised from Dr. Michael McCarthy at CMU, Dr. Cesare Pautasso at ETH, Prof. Andrew Tanenbaum at UA

Web Services

Dr. Minxian Xu

Associate Professor Research Center for Cloud Computing Shenzhen Institute of Advanced Technology, CAS <u>http://www.minxianxu.info/dcp</u>

> 天网疏难漏,世网密莫通。我心久不动,一脱二网中 高竹漱清泉,长松迎清风。此时逢此景,正与此心同。 ——(宋)邵雍





• Q1: Name and explain three transparencies that should be addressed by distributed file systems.











• Q2: What are the advantages and disadvantages of using absolute names as a naming strategy?







Review



- Q3: What are the advantages and disadvantages of the following simple distributed file system?
- RPC to access file system calls
 - No client/local caching





Review



- Q4: What are the advantages and disadvantages of the following simple distributed file system?
- Cache files on clients and perform local file system operations





What is a web service?





From Globus.org (Grid Computing)



With Subs



Ref: Tanenbaum & Van Steen, Distributed Systems: Principles and Paradigms, 2007.







Distributed Systems & Web Services

- Web services provide standards for developing large-scale distributed systems
- One example: "the Grid" is adopting Web services as standard protocols to build a distributed infrastructure for utility based computing
- Web services on the path of success while CORBA distributed objects failed (This is nothing technical, only a matter of widespread industry acceptance)

The	The	Web	Semantic
Internet	WWW	Services	Web
1973	1992	2000	?
Standard Network	Standard User Interface	Standard API	Standard API Metadata



- Distributed systems are built using standardized layers of increasingly higher abstraction levels.
- It took 20 years to go from the TCP/IP (Internet, 1973) standard to the HTTP/HTML (World Wide Web, 1992) standards.
- By reusing HTTP, the time to standardize SOAP/XML was halved (Web Services, 2000).









Layers in Distributed Systems

- **Client** is any user or program that wants to perform an operation over the system. To support a client, the system needs to have a **presentation layer** through which the user can submit operations and obtain a result
- The **application logic** establishes what operations can be performed over the system and how they take place. It takes care of enforcing the business rules and establish the business processes. The application logic can be expressed and implemented in many different ways: constraints, business processes, server with encoded logic ...
- The **resource manager** deals with the organization (storage, indexing, and retrieval) of the data necessary to support the application logic. This is typically a **database** but it can also be a text retrieval system or any other data management system providing querying capabilities and persistence.

Distributing the Layers



• In general, all layers can be distributed







Limitations of Middleware



A direct connection between different organizations is not allowed (security breach) and sometimes not possible (incompatible middleware)



Limitations of Middleware



Conceptually, it could be possible to use a global middleware. However, in practice, there is no "place" for it.



Limitations of Middleware



Point to Point solutions are expensive and do not scale well with the number of systems to be integrated



Web services for integration



Publishing the systems to be integrated as Web Services simplifies the integration and keeps the companies decoupled



- WebGPT: Browser-assisted question-answering with human feedback
- When we want to scale up to huge datasets and models, eventually one machine won't be enough.
- using multiple machines to do learning!

M Tell me how ChatGPT works.

ChatGPT is a large language model that uses deep learning techniques to generate human-like text. It is based on the GPT (Generative Pre-trained Transformer) architecture, which uses a transformer neural network to process and generate text. The model is pre-trained on a massive dataset of text, such as books, articles, and websites, so it can understand the patterns and structure of natural language. When given a prompt or a starting point, the model uses this pre-trained knowledge to generate text that continues the given input in a coherent and natural way.







InstructGPT



- (1) Supervised fine-tuning
- (2) Reward model training
- (3) Reinforcement learning via proximal policy optimization on this reward model.





- Training Data is **Large** 1TB to 1PB
- Complex Models with **Billions** and **Trillions** of Parameters
- Parameters are shared globally among worker nodes:
 - Accessing them incurs large Network costs
 - Sequential ML jobs require **barriers** and hurt performance by blocking
 - At scale, **Fault Tolerance** is required as these jobs run in a cloud environment where machines are unreliable and jobs can be preempted



Key Goals and Features of Design



- Efficient Communication: asynchronous communication model (does not block computation)
- Flexible Consistency Models: Algorithm designer can balance algorithmic convergence and system efficiency
- Elastic Scalability : New nodes can be added without restarting framework
- Fault Tolerance and Durability: Recovery from and repair in 1 sec.
- **Ease of Use**: easy for users to write programs

Distributed Training for Large Models

Communication Patterns

- **Push:** Machine A sends some data to machine B.
- **Pull:** Machine B requests some data from machine B.
- **Broadcast:** Machine A sends data to many machines.
- **Reduce:** Compute some reduction (usually a sum) of data on multiple machines C1, C2, ..., Cn and materialize the result on one machine B.
- All-Reduce: Compute some reduction (usually a sum) of data on multiple machines and materialize the result on all those machines.



Distributed Training for Large Models



- Workers get the Assigned training data
- Workers Pull the Working set of Model
- Iterate until Stop:
 - Workers Compute Gradients
 - Workers **Push** Gradients
 - Servers Aggregate into current model
 - Workers Pull updated model



Large models





Ref: A survey of Large Language models

Small GPTs



- GPT4All
 - Run on single computer



GPT4All is made possible by our compute partner Paperspace.



Run on an M1 Mac (not sped up!)

- NanoGPT
 - repository for training/finetuning medium-sized GPTs.

available GPT implementations



MINGET nanoGPT





Challenges in memory usage

 Many fragments exist in LLM memory usage in distributed systems





PagedAttention Parallel sample



- Prompts can be shared and only one copy is required
- Reference count points to logic block



Physical KV blocks



Centralized scheduler to coordinate distributed GPU worker nodes





Web Services and Grid Computing

SOAP (W3C), WSDL (W3C), UDDI (OASIS), WS Interop(WS-I), Grid (GGF)

SQL/XML	XML Transformations (W3C)	XML APIs
(ANSI & ISO)	XPath, XSL, XSLT, XQuery	DOM (W3C), SAX

XML Vocabularies (OASIS, etc)

Basic XML Constructs (W3C) Canonical XML, XML Fragments, XInclude, XLink, XPointer, XPath

XML Schema and XML Namespaces

XML and DTDs

Unicode (Unicode consortium), URL, http, WebDav (IETF)

Some Important Standards







Web Services

- Provide service interfaces.
- Communicate using request and reply messages made of SOAP or some other XML document.
- Have an Interface Definition Language (IDL) called WSDL (Web Service Definition Language)
- May be looked up in a web service UDDI registry (Universal Directory and Discovery Service).
- Are language independent.
- May be synchronous or asynchronous.

Way To Innovation 34

Web Services







Web Services Infrastructure and Components



Applications				
	Directory service Security Orchestration			
Web Services	Service descriptions (in WSDL)			
SOAP				
URIs (URLs or URNs) XI	ML HTTP, SMTP or other transport			

The Complete Web Services "Stack"






- In general, web services use either a synchronous requestreply pattern of communication with their clients or they communicate by asynchronous messages.
- The client does not block on asynchronous calls. Do you block when you are expecting an important phone call? If not then you are planning on handling the call asynchronously.
- To allow for a variety of patterns, SOAP is based on the packaging of single one-way messages.
- SOAP is used to hold RPC style parameters or entire documents.
- SOAP may be used over different transports (SMTP, TCP, UDP, or HTTP)



- URI's are Uniform Resource Identifiers.
- URL's are Uniform Resource Locator URI's that include location information. Thus, resources pointed to by URL's are hard to move.
- URN's are Uniform Resource Name URI's that include no location information.
- A URN lookup service can be employed to determine a URL from a URN.
- URL's are the most frequently used form of URI.

Examples:

- URL: http://www.siat.ac.cn
- URN: urn:ISBN:0-111-2345-6

Web Service Composition











- By far the most popular Web server is Apache, which is estimated to be used to host approximately 70% of all Web sites.
- Apache's runtime environment, known as the Apache Portable Runtime (APR), is a library that provides a platform-independent interface for file handling, networking, locking, threads, and so on.

There is a hook to translate a URL to a local file name. Such a translation will almost certainly need to be done when processing a request.

Likewise, there is a hook for writing information to a log

A hook for checking a client's identification

A hook for checking access rights

A hook for checking which MIME type the request is related to (e.g., to make sure that the request can be properly handled).









An important problem related to the client-server nature of the Web is that **a Web server can easily become overloaded**. A practical solution employed in many designs is to simply replicate a server on a cluster of servers and use a separate mechanism, such as a **front end**, to redirect client requests to one of the replicas.

A crucial aspect of this organization is the design of the front end as it can become a serious performance bottleneck, what will all the traffic passing through it.





- Defines a scheme for using XML to represent the contents of request and reply messages as well as a scheme for the communication of XML documents.
- It is intended that a SOAP message can be passed via intermediaries on the way to the computer that manages the resources to be accessed.
- The intermediaries may process the SOAP to provide security or transaction support as well as other services.
- Typically, the SOAP header is processed by intermediaries and the SOAP body holds the request or reply.

SOAP Envelope









In this figure and the next, each XML element is represented by a shaded box with its name in italic followed by any attributes and its content







 POST /examples/stringer
 endpoint address

 Host: www.cdk4.net
 endpoint address

 Content-Type: application/soap+xml
 action

 Action: http://www.cdk4.net/examples/stringer#exchange
 action

 <env:envelope xmlns:env=namespace URI for SOAP envelope>
 env:header>

 <env:header> </env:header>
 <env:body> </env:body>

 </env:Envelope>
 endpoint address

A transport protocol is required to send a SOAP document to its destination.

Other transports may be used. WS-Addressing may be used to include destination and source. Thus, different protocols might be used over different parts of the route of a message.



<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"> <S:Header> <wsa:MessageID> uuid:6B29FC40-CA47-1067-B31D-00DD010662DA </wsa:MessageID> <wsa:ReplyTo> <wsa:Address>http://business456.example/client1</wsa:Address> </wsa:ReplyTo> <wsa:To>http://fabrikam123.example/Purchasing</wsa:To> <wsa:Action>http://fabrikam123.example/SubmitPO</wsa:Action> </S:Header> <S:Body>

</S:Body> </S:Envelope> Address information included within the document rather than only being specified by the transport.



At first glance, the interaction between client and server seems like RMI.

But, RMI permits the creation of **remote objects**. These may then be accessed via remote references.

Web services may create and use objects but never return a remote reference to a remote object. A web service is a single object that offers a set of procedures.





- The primary means of describing a web service is by using WSDL (the Web Services Description Language)
- XML Schema may be used to describe documents.
- WSDL makes use of XML Schema to describe an exchange of messages.
- A Service Description (WSDL document) is an IDL plus it contains information on how and where the service may be accessed.
- It contains an abstract part and a concrete part. The abstract part is most like a traditional interface. The concrete part tells us how and where to access the service.



A binding is a choice of protocols.

A service holds an endpoint address.

Client or server side code may be generated automatically from the WSDL. A WSDL document may be accessed directly or indirectly through a registry like UDDI (Universal Directory and Discovery Service).



Name	Messages sent by			
	Client	Server	Delivery	Fault message
In-Out	Request	Reply		may replace <i>Reply</i>
In-Only	Request			no fault message
Robust In-Only	Request		guaranteed	may be sent
Out-In	Reply	Request		may replace Reply
Out-Only		Request		no fault message
Robust Out-Only	I	Request	guaranteed	may send fault



- XSDL (The XML Schema Definition Language) allows us to describe the structure of an XML message
- WSDL allows us to describe message exchanges





- A message exchange is called an operation
- Related operations are grouped into *interfaces*
- A *binding* specifies concrete details about what goes on the wire



WSDL



- Describes the contract between applications
- Can be automatically generated from a collection of Java or C# classes
- Can be read by utilities that generate client side proxy code or server side skeletons.
- See wsimport (JDK 6.0) or wsdl.exe on the Microsoft side



<!- abstract definitions \rightarrow

<types>

- <messages>
- <portType>
- <!- concrete definitions \rightarrow
- <binding>
- <service>
- </definition>





<!- Terms found in application code \rightarrow

<types>

<messages>

<portType>

- <!-- Handled by XML infrastructure \rightarrow
- <binding>

<service>

</definition>





<types>

- a container for XSDL Type definitions
 - element names may be defined here as well



WSDL Structure



<definition>

<types>

For example, in Google's WSDL, GoogleSearchResult is defined as a complex type with many elements.





- <types>
- <message>
 - May have more than one part (think parameters)
 - Define the input or output of an operation
 - RPC style messages associate a name with a type (defined above)
 - Document style messages associate a name with an XML element

</definition>



<definition> <types> <message> Two examples: - In Google's WSDL, a doGoogleSearch message is defined with many parts of basic xsd types. - In Google's WSDL, a doGoogleSearchResponse message is defined as of type GoogleSearchResult </definition>



<definition> <types>

- <messages>
- <portType>
 - The definition of an interface or group of operations
 - The term "portType" will be replaced with the term "interface" in WSDL 1.2
 - Each operation has a name and normally specifies both input and output messages

</definition>



<definition> <types> <messages> <portType>

- For example, in Google's
 WSDL, GoogleSearchPort contains three operations.
- The operation doGoogleSearch has an input message (doGoogleSearch) and an output message (doGoogleSearchResponse.)

</definition>



<definition> <types> <messages> <portType> <binding>

- Each binding has a unique name that is associated with a particular interface.
- The protocol used is specified.
- Details found here specify how the data will look on the wire.

</definition>



<definition> <types> <messages> <portType> <binding>

- For example, in Google's WSDL, the binding name GoogleSearchBinding is introduced and is associated with the interface GoogleSearchPort.
- Each operation within that interface is described as soap operations.

</definition>



<definition> <types><messages><portType> <binding> <service>

- Defines a collection of ports (endpoints) that exposes a particular binding
- An address is associated with a binding

</definition>



<types><messages><portType><binding>

<service>

For example, in Google's WSDL, the service name GoogleSearchService is introduced. The interface GoogleSearchPort is associated with the binding GoogleSearchBinding.

The service element holds the address of the service.

</definition>



- (1) Get the WSDL from http://www.google.com/apis/
- (2) If using .NET run wsdl.exe on GoogleSearch.wsdl.
- (3) If using Java and Axis run wsdl2java.bat on GoogleSearch.wsdl.
- (4) wsdl2java.bat holds the line

java org.apache.axis.wsdl.WSDL2Java %1 The WSDL2Java class is in axis.jar



A Google Client in Java



// Running a simple Google RPC client for spell checking

import GoogleSearch.*; // wsdl2java generated package

public class MyGoogleClient{

private static String endpointAddress = "http://api.google.com/search/beta2";

public static void main(String[] args) throws Exception {

```
if(args.length != 1) {
    System.out.println("Usage1: java MyGoogleClient wordToSpellCheck");
    System.out.println(
    "Usage2: java MyGoogleClient \"a phrase to spell check\"");
    System.exit(0);
```

}



System.out.println("Contacting Google Web Service at " + endpointAddress); System.out.println("Checking on spelling of '" + args[0]+"'");

```
GoogleSearchServiceLocator loc = new GoogleSearchServiceLocator();
```

```
GoogleSearchPort gp = loc.getGoogleSearchPort();
```

```
String answer = gp.doSpellingSuggestion(
"n6IHU/FQFHIHzpbzRTPFvrUP4Cw+/k+N",
args[0]);
```

if(answer == null) System.out.println("Google likes the spelling of '" + args[0]+"'"); else System.out.println("Google suggests the spelling '" + answer +"'");



GoogleSpring2005\java>java MyGoogleClient "Cornegi Melon Universeti" Contacting Google Web Service at http://api.google.com/search/beta2 Checking on spelling of 'Cornegi Melon Universeti'

Google suggests the spelling 'Carnegie Mellon University'


A Google Client in C#



// run a client against Google's web service
using System;

```
namespace ConsoleApp
```

{

{

```
class GoogleClient
```

```
public static void Main(string[] args) {
```

try {
 GoogleSearchService s =
 new GoogleSearchService();





```
Console.WriteLine("Enter word to spell check");

String word = Console.ReadLine();

String answer = s.doSpellingSuggestion(

"n6IHU/FQFHIHzpbzRTPFvrUP4Cw+/k+N", word);

Console.WriteLine("Google returned " + answer);

}

catch(Exception ) {Console.WriteLine("Google threw an exception");}
```





•Client-side caching generally occurs at two places:

1. Most browsers are equipped with a **simple caching facility**. Whenever a document is fetched it is stored in the browser's cache from where it is loaded the next time. Clients can generally configure caching by indicating when consistency checking should take place.

2. A client's site often runs a Web proxy. As we explained, a Web proxy accepts requests from local clients and passes these to Web servers. When a response comes in, the result is passed to the client.

•The **advantage** of this approach is that the proxy can cache the result and return that result to another client, if necessary. In other words, a Web proxy can implement a **shared cache**



•In addition to caching at browsers and proxies, it is also possible to place caches that cover a region, or even a country, thus leading to **Hierarchical caches**. Such schemes are mainly used **to reduce network traffic**, but have the **disadvantage** of **potentially incurring a higher latency compared to using non-hierarchical schemes**.





- An acronym for Universal Directory and Discovery Services.
- A directory service for use with web services.
- One way to obtain service descriptions.
- May be used within organizations to perform lookups for WSDL documents.
- Supports white pages (lookup by name) and yellow pages (lookup by attribute)
- Provides a publish/subscribe interface.
- Uses replication among many servers for scalability.
- JAXR (The Java API for XML Registries) may be used to interact with UDDI.







XML Web Services Security SAML (Security Assertion ML),XKMS (XML Key Management Specification), XACML (eXtensible Access Control Markup Language)

XMLDSIG (W3C) XMLENC (W3C)

.NET Crypto API's

Java Security API's

Travel Agent Scenario



- 1. The client asks the travel agent service for information about a set of services; for example, flights, car hire and hotel bookings.
- 2. The travel agent service collects prices and availability information and sends it to the client, which chooses one of the following on behalf of the user:

(a) refine the query, possibly involving more providers to get more information, then repeat step 2;

(b) make reservations;

(c) quit.

3. The client requests a reservation and the travel agent service checks availability.

4. Either all are available;

or for services that are not available;

either alternatives are offered to the client who goes back to step 3; or the client goes back to step 1.

5. Take deposit.

- 6. Give the client a reservation number as a confirmation.
- 7. During the period until the final payment, the client may modify or cancel reservations
 The Business Process Execution

The Business Process Execution Language (BPEL) is used to write such scenarios in XML.

Case Study: The Grid

- Grid refers to middleware that is designed to allow for sharing of resources such as data and CPU cycles on a very large scale.
- Provides for heterogeneity, management, and security.
- Latest version runs over web services.
- The open source Globus Toolkit implements the grid architecture.
- The immense quantity of data in archives makes ftp or web access infeasible.





- Move around the sky by clicking and dragging.
- Zoom in/out by scrolling the mouse wheel or pressing +/-.
- Right-click in the view to display the Finder Scope for more information.
- Menu tabs ("Explore", "Guided Tours", etc.) have two parts. Click the tab's top to open
 a pane; click the tab's bottom to open a submenu.







Some Grid Projects



Description of the project	Reference
1. Aircraft engine maintenance using fault histories and sensors for predictive diagnostics	www.cs.york.ac.uk/dame
2. Telepresence for predicting the effects of earthquakes on buildings, using simulations and test sites	www.neesgrid.org
3. Bio-medical informatics network providing researchers with access to experiments and visualizations of results	nbcr.sdsc.edu
4. Analysis of data from the CMS high energy particle detector at CERN by physicists world-wide over 15 years	www.uscms.org
5. Testing the effects of candidate drug molecules for their effect on the activity of a protein, by performing parallel computations using idle desktop computers	[Taufer et al. 2003] [Chien 2004
6. Use of the Sun Grid Engine to enhance aerial photographs by using spare capacity on a cluster of web servers	www.globexplorer.com
7. The butterfly Grid supports multiplayer games for very large numbers of players on the internet over the Globus toolkit	www.butterfly.net
8. The Access Grid supports the needs of small group collaboration, for example by providing shared workspaces	www.accessgrid.org