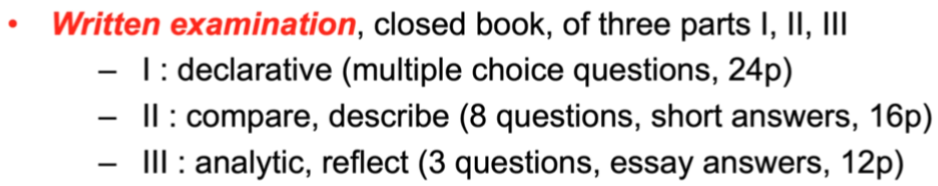
ID2201 Review Summary

1. Course Overview

1. **Introduction** - what is a distributed system, and why is it different [Chapters 1 and 2]
2. **Erlang** - concurrent and distributed programming in Erlang.
3. **Networks and process communication** - things you might (or should) know, but we'll go through them again [Chapters 3 and 4]
4. **Remote invocation** - language constructs to program distributed systems [Chapter 5]
5. **Indirect Communication** - group communication, publish/subscribe, and message queue systems [Chapter 6]
6. **File systems and Name services** - the problems of a distributed file system, performance, consistency [Chapters 12 and 13]
7. **Time** - a simple thing that turns out to be very complex [Chapter 14.1-4]
8. **Global state** - can we describe the state of a distributed system and what can we determine [Chapter 14.5]
9. **Coordination and agreement** - how do we agree, and how do we know that we agree? [Chapter 15]
10. **Transactions** - how can we make a set of operations behave as an atomic operation? [Chapter 16]
11. **Distributed transactions** - now how do we solve it if we have multiple servers [Chapter 17]
12. **Replication** - building fault-tolerant systems [Chapter 18]

TEN1 - An approved written exam graded A-F

- **A proctored computer-based closed-book exam in Canvas**
- **Multiple Choice, Multiple Answer, True/False, and Numeric**



- ***Written examination***, closed book, of three parts I, II, III
  - I : declarative (multiple choice questions, 24p)
  - II : compare, describe (8 questions, short answers, 16p)
  - III : analytic, reflect (3 questions, essay answers, 12p)

2. Introduction

Definition of Distributed System:

"One in which hardware and software components located at networked computers communicate and coordinate their actions only by message passing"

Motivation:

- Resource Sharing
- Communicate
- Geographically distributed: Data, Computers, Resources, Clients

- Performance, Scalability, availability, fault tolerance

Applications and Services

- Printer Servers, Distributed File Systems[DNS], DNS, ssh
- WWW: Web servers/Browsers, FTP and Mail Servers/Clients, Instant Messaging, Online Games, CDNs, Streaming Media Applications
- E-commerce, Banking
- Remote Control and Monitoring
- Scientific and Engineering Computing
- Social Networks

Major Aspects, Features, Problems

Distribution, Concurrency, Communication, Messages, Time, Security, Coordination, Failures

There is <u>no Shared Memory</u> but <u>only Message Passing</u> in Distributed System.

Messages:

- Encoding, Marshaling, Unmarshaling, reconstruct data structure

Multipurpose Internet Mail Extensions - MIME

- An Internet standard that extends the format of email to support
- Text in character sets other than ASCII
- Non-text attachments: audio, video, images, application programs
- Message bodies with multiple parts.

Coordination:

- Two generals problem (lunchtime problem): We don't know how long it takes for a message to be delivered.

Failure

- Monitor if a interactive thread is Dead or Alive

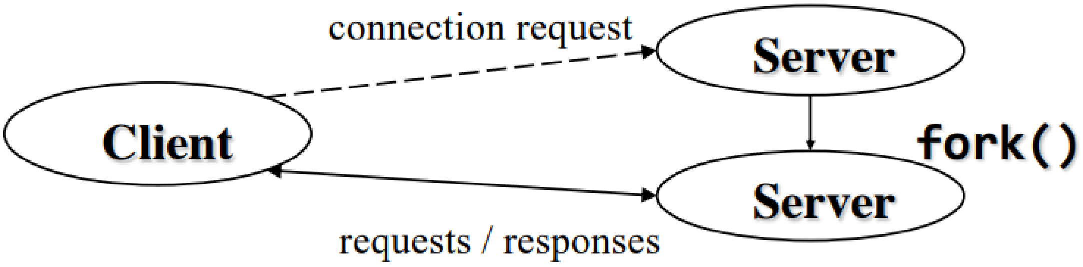
Distributed System Features

- Time - Two processes can't agree on the time (real-time)
- Coordination, Failure Detection

Basic Architectures

1. **2-Tier Client-Server Architecture**

   The **client** is the entity (process) accessing the remote resource, and the **server** provides access to the resource.

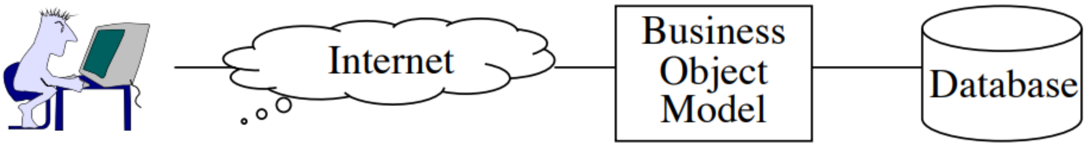


Problems:

- Portability - 可移植性，不同平台兼容不足
- Efficiency and scalability - 客户增加，服务器负载过重
- Fault Tolerance (Single Point of Failure) - 一个点故障就崩溃，没有冗余
- Security

2. **3-Tiered Architecture**

   - User- Interface Tier
   - Business Logic Middle Tier: inventory control, budget, transaction monitors, ORBs, authentication
   - System Service Tier: 数据层，与数据库交互



Improved:

- Faster Protocols than HTTP
- **"Thiner"** Client GUI
- Middle tier control user authentication
- Server can keep User Data

3. **Peer-to-Peer (P2P) Architecture**

   - Peers run on an overlay network. Equal in **Responsibility, Capabilities, Functionality**.
   - Overlay Network: virtual network of nodes created on top of and existing network, internet.
   - Each node has an ID, knows neighbors, does not know the global topology, communicates as a source and a destination, and serves as a router sending data.
   - Distributed Hash-Table (DHT)

4. **Service-Oriented Architecture (SOA)**

3.1 Networks and Interprocess Communication

Reliability, Security, Performance, Ability to meet Timeliness Guarantees, Guaranteed Bandwidth, Bounded Latencies for Communication Channels

The Internet is the largest internet that includes commercial, military, university, and other networks with different physical links and various protocols, including IP (Internet Protocol)

- WAN（Wide Area Network，广域网）大范围地理区域
- MAN（Metropolitan Area Network，城域网）校园网，企业网
- LAN（Local Area Network，局域网）家庭Wi-Fi
- PAN（Personal Area Network，个人区域网络）个人设备，蓝牙

Latency

- Distance - speed of signal
- Access - granting of resource
- Routing - processing in nodes

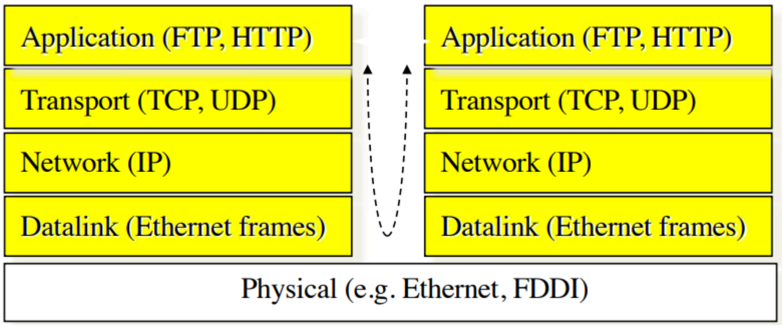
Ping

- Used to test the reachability of a host on an Internet Protocol network.
- Ping measures the round-trip time.
- Operates by sending Internet Control Message Protocol (ICMP) echo request packets to the target host and waiting for an ICMP echo reply.

Packet Delivery Time (Latency) = Transmission Time + Propagation Delay

- **Packet Delivery Time:** The first bit leaves the transmitter until the last is received.
- **Transmission Time** = Packet size / Bit rate
- **Propagation time** = Distance / Propagation speed

Multi-Layered Network



- **Application** - The end product
- **Presentation** - Encoding information, Serialization序列化, Marchaling编组
- **Session** - Security, Authentication, Initialization
- **Transport** - Messages, Streams, Reliability, Flow Control
- **Network** - Addressing of nodes in a network, Routing, Switching
- **Data link** - Point to point deliver of frames, Medium access, Link Control

- **Physical Layer** - Bits of analog signals, electrical, optical, radio ...

HTTP, FTP,SMTP, TCP, UDP, IP, ARP, Ethernet, Wi-Fi....

ICMP: Internet Control Message Protocol

SCTP: Stream Control Transmission Protocol

Routing:

1. **Distance Vector距离向量**: Sending Routing Table to neighbors, RIP, BGP

   只与邻居交换信息，不广播。拓扑变化时，更新路由表速度慢，大规模网络效率低。适合小型网络。

   Slow convergence.

   - Routing Information Protocol (RIP): 路由信息协议Employs the hop count as a routing metric.

     Send a routing table to neighbors each 30 sec.

   - Border Gateway Protocol (BGP): 边界网关协议 在AS自治系统之间交换路由信息。使用复杂的路由策略和属性（如路径长度、网络策略、路由器配置）来选择最佳路径。适合处理规模庞大的互联网。

2. **Link State链路状态**: Tell everyone about your direct links, OSPF

   拓扑变化，更新快，适合动态网络。每个路由器维护全网拓扑，计算开销大。适合大型网络。

   - Open Shortest Path First (OSPF)

IP Address

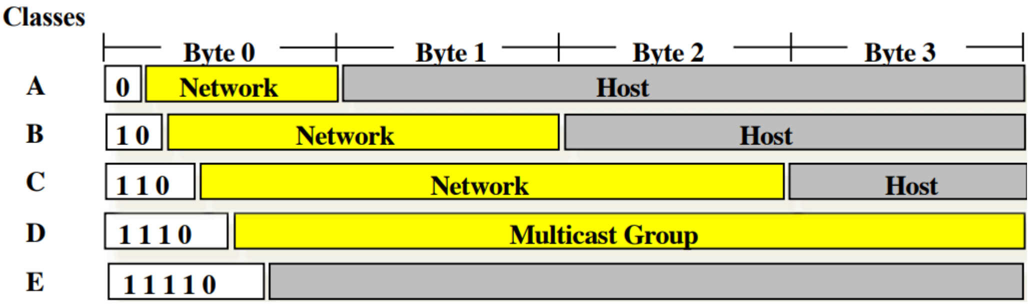
Class A : 1.0.0.0-126.255.255.255 大型网络

Class B: 128 - 191 中型网络

Class C: 192 - 223 Wi-Fi

Class D: 224 - 239 Multicasting

Class E: 240 - 255



Classful Routing

- Five class A-E, Obsolete 过时的
- Three parts: **Network, Subnet, and Host**

Classless Routing

- Two parts: **Subnet and Host**, 根据实际的主机数量灵活分配子网大小
- `192.168.1.0/24` 表示前24位是网络ID，后8位是主机ID

TCP - Transmission Control Protocol: Stream

- Duplex stream abstraction, Flow Control, Congestion friendly 阻塞友好, slows down if a router is choked 阻塞
- 连接，Reliability (Lost / Erroneous Packets Retransmitted)，按顺序。三次握手。
- 文件传输，电子邮件。

UDP - User Datagram Protocol: Datagram

- Datagram Abstraction, Independent Messages, Limited in Size
- Low Cost, No Set Up / Tear Down Phase
- No Acknowledgment
- 无连接，不用确认，不保证数据包的顺序，可靠。
- 开销延迟低，适合在线游戏，视频流。使用端口号

ICMP - Internet Control Message Protocol

- A requested service is not available / A host or router could not be reached
- 发送错误信息/网络状态信息, "ping"命令

IGMP - Internet Group Management Protocol

- On IPv4 networks to establish multicast group memberships. IGMP is an integral part of IP multicast. 管理 IPv4 网络中的多播组，视频广播。

RSVP - Resource Reservation Protocol

- 为集成服务互联网在网络上预留资源，实时数据流（如视频会议）

SCTP - Stream Control Transmission Protocol

- 传输层，类似 TCP 和 UDP，更复杂的需求（如电话或视频会议），多条流和消息顺序。
- 比 TCP 更适合需要高可用性和复杂数据传输的场景。

Sockets 网络层，代表网络连接的端点。

- 网络编程的基础工具，用来在不同计算机或进程之间建立通信。
- 用**IP Address**， **端口号**， **TCP/UDP** 标识
- **Stream Socket:**
  - **Server** - **Creates** a listening socket bound to a port (**create**, **bind**, **listen**)

    Accepts an incoming connection request and **creates** a communication socket for **reading**/**writing** a byte stream.

    - **Client** - **Create** a communication socket and **connects** it to a server identified by an IP address and a port. **Reads**/**writes** from a socket.
- **Datagram Socket:**
    - **Server** – **Create** a message socket and **bind** it to a port. - **Receive** an incoming message (message contains a source IP address and port number).
    - **Client** - **Create** a message socket bound to a source port. – **Create** a message and **give** it a destination address and port number. – **Send** the message.

Marchaling Data - Transform Internal Data Structure into the Sequencing of Bytes

- Java **Serialization** 接口, Erlang **external term format**
- **Independent**: XML, Google Protocol Buffer, ASN.1
    - Message format defined by specification: XML Schema, .proto, ...
    - A compiler uses the specification to generate an encoder and decoder

在理想的世界里，应用层应该与底层网络的实现无关，但现实中开发分布式应用需要对网络的延迟、带宽、可靠性、丢包率等特性有深入的理解。只有这样，才能优化应用的性能、提升用户体验，并确保应用在不同网络环境下都能稳定运行。

3.2 MPI: Message Passing Interface

多处理器、集群和异构网络的消息传递库规范，API（应用编程接口）的规范。

Feature:

- A message-passing **library specification** for multiprocessors, clusters, and heterogeneous networks
- Designed to allow the development of **parallel software libraries**
- To provide **access to advanced parallel hardware** for end users, library writers, and tool developers

MPICH - 一种实现， 多种平台运行

LAM - Pour TCP/IP Networks

MPICH - Another message-passing programming environment

主要功能:

- **点对点通信（Point-to-point）**
- **集体通信（Collective communication）**：在多个进程之间进行同步通信，例如广播、聚合等操作。
- **进程组（Process Groups）**：MPI 管理并发的进程组，可以让程序组织和管理多个并行进程。
- **拓扑结构（Topologies）**：为进程之间的通信建立特定的拓扑结构，例如网格、环形等。

MPI 进程是并行计算中的基本单位，进程通过**消息传递**在**通信器**内进行同步或异步通信。

---

4. Remote Invocation

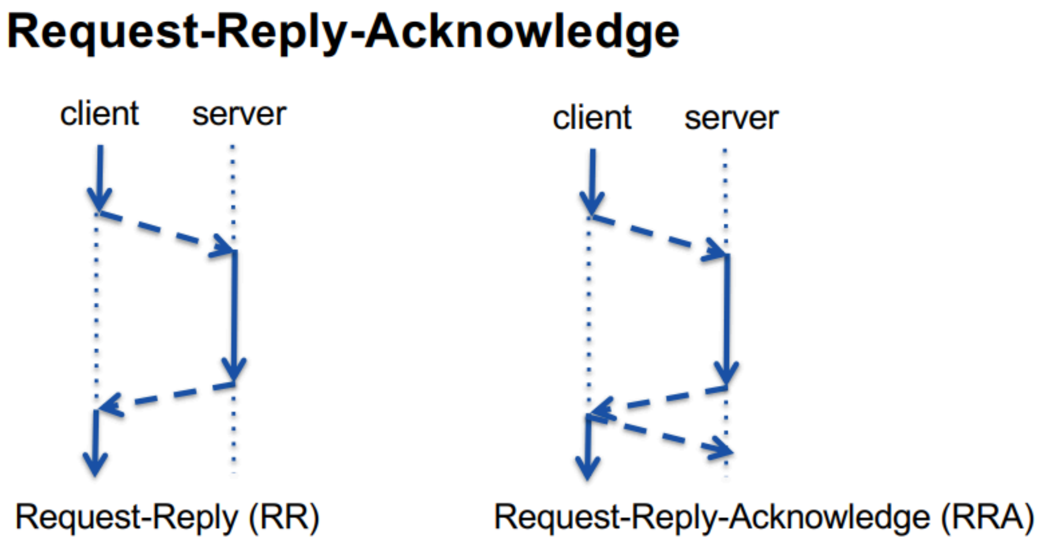
Idempotent Operation(幂等操作)

- Can be performed **repeatedly** with the **same effect** as if it had been performed exactly once
- For example, add an element to a set. It will always have the same effect on the set each time it is performed.

Request Protocol

Request-Reply Protocol

Request-Reply-Acknowledge reply Protocol



At-most-once（一种执行模型）：

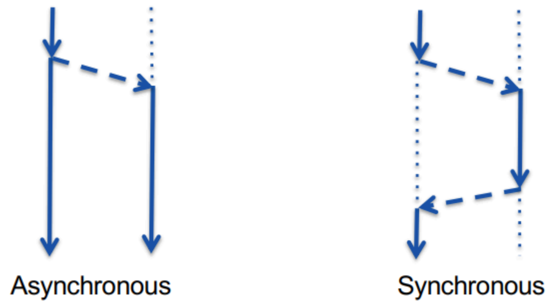
- The request has been executed **once**. Implemented using a history or simply not resending requests.
- **Non-Idempotent Operations** 因为每次都会改变
- **No re-sending** requests, **simple**, **not fault tolerant**
- With history: Expensive to implement, fault-tolerant

At-least-once:

- The request has been executed **at least once**. No need for a history; simply resend requests until a reply is received.
- **Idempotent Operations** 幂等操作
- Simple to implement, **fault-tolerant**

***以上两种情况，如果客户端没收到reply，客户端也没法确定request有没有被执行**

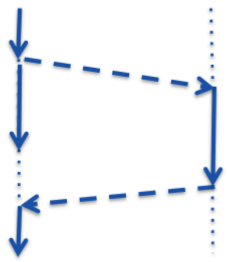
Synchronous or Asynchronous



Continue to execute - 在发送请求后，客户端不会等待服务器的立即回复，而是继续执行其他操作。

Suspend if not arrived - 客户端在需要回复时检查是否已经收到服务器的响应。如果还没收到，则客户端会暂停当前相关任务，直到回复到达。

RR over Asynchronous



- send request
- continue to execute
- suspend if not arrived
- read reply

HTTP

```
Request = Request-Line *(header CRLF) CRLF [message-body]
Request-Line = Method SP Request-URI SP HTTP-Version CRLF
```

E.g. 请求方法， 目标资源URL, 协议版本， 回车换行符。（请求头）。空行。请求体。

```
GET /index.html HTTP/1.1\r\n foo 42 \r\n\r\nHello
```

Methods: GET, HEAD, POST, PUT, DELETE.

HTTP - Hypertext Transfer Protocol:

A general-purpose request-reply protocol.

REST - Representational State Transfer

- XML, JSON
- Lightweight, 简单HTTP请求

SOAP - Simple Object Access Protocol

- HTTP, SMTP, 格式：SOAP, XML
- Standardized标准化, Heavyweight重量级

RPC (Remote Procedure Call):

- 允许程序调用不同计算机上的函数， **Synchronous Operation**
- Program Number 一组相关的远程过程；**Version Number** 远程程序的版本；**Produce Number** 一个程序中不同的过程号
- Three Unsigned Fields: **Remote Program Number / Program Version Number / Procedure Number**
- Server will initialize **different concurrent processes**, might need **synchronization**

RMI (Remote Method Invocation)

Object-Oriented Analog of RPC，调用远程对象的方法，不用关心对象在哪，Java RMI

Procedure Call - 远程调用

- 隐藏底层网络通信的复杂性
- Find the procedure 定位具体函数
- Give the procedure access to arguments 将参数传输过去
- Pass control to the procedure 传递控制权，执行代码
- Collect the reply if any 收集返回结果
- Continue execution 继续执行

Call by Value / Reference 按值传递 / 按引用传递

当按值传递一个引用时：

- 你可以通过该引用访问和修改原始数据（因为引用仍然指向相同的数据）
- 你不能通过该引用修改原始引用本身的指向（因为你修改的只是副本）

ONC - Open Network Computing RPC

- 内部网络 intranet
- At - least - once call semantics
- Interface Definition Language - IDL
- XDR 外部数据表示法 + UDP

Java RMI 一种面向对象的RPC

- Invoke Methods of Remote Objects
- At - most - once 不会重复执行
- Pass by value 副本 / By reference 指向真实对象
- Remote Object: Reference   Serializable Object: Value

RPC allows calling procedures over a network; RMI invokes objects’ methods over a network.

Location transparency: invoke a method on a stub like on a local object

Location awareness: the stub makes remote calls across a network and returns results via stack

Stub: 存根对象， 是远程对象的代理， 被调用时通过网络发起远程调用，传参，传结果。

Naming Service

- Object's unique name. Bind the name. 一种将远程对象与唯一名称关联的机制
- **命名服务的定位**是一个核心问题，通常通过配置命名服务的 URL 来解决

Examples: 远程调用的不同机制

- **SunRPC**: Call-by-value, At-least-once, IDL, XDR, Binder
- **JavaRMI**: Call-by-value/reference, At-most-once, Interface, JRMP (Java Remote Method Protocol), Rmi Registry
- **Erlang**: Message Passing, Maybe, No, ETF (External Term Format), Local Registry Only
- **CORBA** (Common Object Request Broker Architecture): Call-by-Reference, IDL, ORB (Object Request Broker), Name Service
- **Web Services**: WSDL (Web Services Description Language), UDDI (Universal Description, Discovery, and Integration)