COMS3200

COMS3200/7201 Week 2 - Interprocess communication

School of Information Technology and Electrical Engineering
The University of Queensland

COMS3200

This week's outline

- Some admin…
- Inter-process communication
  - Message passing
  - Remote procedure calls
- Data representation

COMS3200

Admin

- Tutorials start this week

Additional material posted to web

- Additional RPC material - extract from Tanenbaum 2nd edition

COMS3200

Last week…

- Network structure
  - Point-to-point
  - Broadcast
- Protocols
  - Protocol hierarchies
- Headers and Enveloping
- Services
  - Service primitives
- OSI and TCP/IP models

COMS3200

Quick Quiz

(To encourage you to recall and consolidate last week's lecture)

- Draw examples of **point-to-point** and **broadcast** network **topologies**.
- How do **peer entities** communicate?
- What's the difference between a **service** and a **protocol**?

You have 2 minutes

COMS3200

Expected Learning Outcomes

- After this week's lecture(s), associated readings and Tutorial 2, you should…
  - Be able to describe the message passing and RPC paradigms of inter-process communication and be able to choose appropriate methods for given distributed applications
  - Understand the issues involved in network data representation and various approaches to solving the problem

COMS3200

Interprocess Communication (IPC)

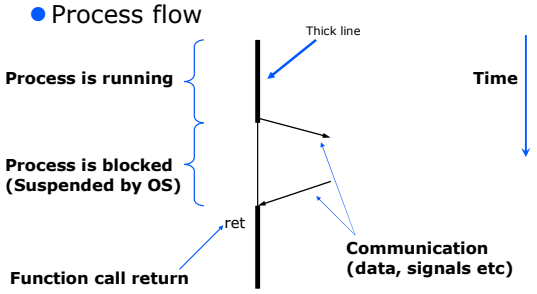
- IPC
  - Processes need to communicate
- Four IPC paradigms
  - Shared memory
  - Message passing
  - Remote procedure calls (RPC)
  - Transactions

7

COMS3200

Notation

- Process flow



8

COMS3200

Message Passing

- Two primitives
  - Send
  - Receive
- Options
  - Synchronous (blocking)
    - Process blocked by OS until communication finished
  - Asynchronous (non-blocking)

9

COMS3200

Message Passing Primitives

- Figures to be drawn in class
  - Blocking send
  - Non-blocking send
  - Blocking receive
  - Non-blocking receive

10

COMS3200

Example

- Choice of IPC primitives
- (To be worked through in class)

11

COMS3200

Client-Server Model Remote Procedure Call

- RPC provides a higher abstraction of communication as message passing
  - Represents communication as procedure invocation
    - hides communication from programmer
    - replaces send and following receive with procedure call
  - Is implemented using message passing

12

COMS3200

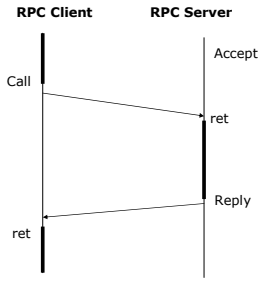
Client-Server Model Remote Procedure Calls

- General Client-server model
  - Figure to be drawn in class
- Remote Procedure Call (RPC) model
  - Figures to be drawn in class
- RPC
  - very popular communication and distributed computing model
  - Design Issues
    - Binding, Heterogeneity, Transparency, Parallelism

T-526 to 529, PD397-399

COMS3200

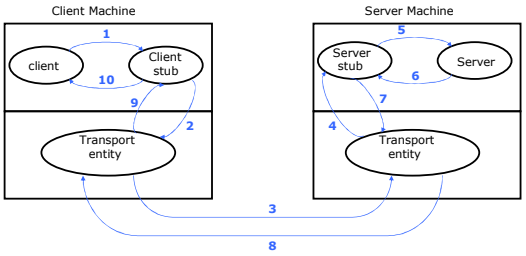
Client-Server Model Remote Procedure Call



COMS3200

Client-Server Model Remote Procedure Call

- Execution of RPC



COMS3200

Client-Server Model Remote Procedure Call

- Design Issues
  - Binding
  - Heterogeneity
  - Transparency
  - Parallelism

COMS3200

RPC Design Issues 1: Binding

- How does caller name and *find* procedure to be called?
- Options
  - Off-line
  - Run-time – dynamic linker
    - (Figure to be drawn in class)

COMS3200

RPC Design Issues 2: Heterogeneity

- How does system deal with
  - multiple machine types
  - programs written in different languages
- Static declarations of procedure interfaces
  - Types must be independent of any programming language
  - Stubs
    - automatically generated
    - perform conversions as needed

RPC Design Issues 3: Parallelism

- How to achieve concurrent operation of client and server?
  - Non-blocking calls
  - Blocking calls with multiple threads

19

RPC Design Issues 4: Transparency

- How much like a local procedure call?
- Problems
  - Client or server failures
    - ➔Exception handling
  - Parameter passing
    - Problems with
      - Call by reference
      - Pointers
      - Procedures and functions
    - ➔Marshalling

20

Exception Handling 1

- **Server crash**, client can
  - hang
  - time-out and report exception
  - time-out and retransmit
- Ideally – **idempotent** operations
- Semantic classifications of an RPC system:
  - Exactly once
  - At most once
  - At least once

21

Exception Handling 2

- **Client crash**
  - leaves orphan RPCs executing on server
- Options are
  - Extermination
  - Expiration
  - Reincarnation
  - Gentle reincarnation

22

Data Representation

- Receiver should extract the same message that the sender sent
  - Sender and receiver must agree to a message format or presentation format

PD-528

23

Exercise...

- How might the (decimal) integer number 1024 be transmitted?
  - ($1024 = 10000000000_2$)
- What about –1?

24

Big-Endian vs Little Endian

Big-endian
- Most-significant byte stored or transmitted first (big end first)

Little-endian
- Least-significant byte stored or transmitted first

Example…

Marshalling

- Argument encoding sometimes called marshalling
  - Decoding -> unmarshalling
- 3 issues
  - Data types
  - Conversion
  - Tagging

Data Types

- What data types might we want to transfer?

Conversion

- Two strategies
  - Canonical intermediate format
    - Particular format defined for each type
  - Receiver-makes-right
    - Sender uses own representation, receiver does any necessary translation

Decoding

- How does the receiver know what's in the message?
  - Example: |03|3c|b7|5b|4b|53|41|4c| (Hexadecimal)
- What could this mean?

Tagging

- Tag
  - Additional information included in message to aid decoding
- Untagged data
  - Receiver just knows
    - How?
- Always need tags for variable length arrays

Data Representation Examples: XDR

- External Data Representation
- Characteristics
  - C types
  - Canonical intermediate form
    - Big-endian, two's complement
  - No tags
    - Except for array lengths

PD-535 to 537 31

Data Representation Examples: ASN.1

- Abstract Syntax Notation
  - ISO standard
- Characteristics
  - C types
  - Canonical intermediate form
  - Tags
- Each data item: <tag, length, value>

PD-537 to 538 32

Data Representation Example: NDR

- Network Data Representation
- Characteristics
  - Receiver-makes-right
  - Each message has architecture tag
    - Integers – big-endian or little
    - Characters – ASCII, EBCDIC
    - Floating Point – IEEE754, VAX, Cray, IBM
  - Individual data items untagged
  - C type system

PD-539 33

XDR Examples

- To be presented in class

34

More Information

- XDR Standard – RFC 1832
  - http://www.faqs.org/rfcs/rfc1832.html
- Big-endian vs Little-endian
  - http://www.NovelTheory.com/techpapers/endian.asp

35

Summary: This Week's Topics

- Inter-process communication
  - Message passing
  - Remote procedure calls
- Data representation

36

COMS3200

What YOU should do in the next week

- Do readings for week 2 (if haven't already)
  - Tanenbaum pages 526-529
- Read additional RPC material
- Attempt Tutorial 2
- Readings for week 3
  - Tanenbaum pages 524-526 (UDP)
  - Tanenbaum pages 532-553 (TCP)

37