

Client-Server Model Remote Procedure Call

- Design Issues
 - Binding
 - Heterogeneity
 - Transparency
 - Parallelism

25

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)

26

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

27

RPC Design Issues 3: Parallelism

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

28

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

29

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

30

Exception Handling 2

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

31

Blocking semantics – summary (1)

- Blocking semantics (blocking send, RPC/RMI) are suitable for client-server communication
 - Unless client has some processing to do when waiting for server's response
- Blocking semantics applied for server-to-server communication negatively impacts on the system performance
 - Unless threads are used (i.e. threads are blocked not the main process)

32

Blocking semantics – summary (2)

- Remote Method Invocation (RMI) in object oriented languages like Java also has blocking semantics so its impact on performance is the same as RPC if applied to server-to-server communication
- RMI has lighter restrictions on parameter types that can be used by method invocation than in RPC

33

Data Representation

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

34

Exercise...

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

35

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

36

Marshalling

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

37

Data Types

- What data types might we want to transfer?

38

Conversion

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

39

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?

40

Tagging

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

41

Data Representation Examples: XDR

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

42

Data Representation Examples: ASN.1

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

43

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

44

XDR Examples

- To be presented in class

45

More Information

- XDR Standard – RFC 1832
 - <http://www.ietf.org/rfc/rfc1832.html>
- Big-endian vs Little-endian
 - <http://www.NovellTheory.com/techpapers/ndian.asp>

46

Summary: This Week's Topics

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

47

What YOU should do in the next week

- Do readings for week 2 (if haven't already)
 - Tanenbaum pg 5th :495-506, 543-546 (4th Ed: 368-376, 526-529)
- Read additional RPC material
- Attempt Tutorial 2
- Readings for week 3
 - Tanenbaum pages 5th :541-543 (UDP) 5th :552-581 (TCP)
 - 4th : 524-526 (UDP); 532-553 (TCP)

48