Wire Formats & Binding for Distributed Object Systems

Bill Janssen

Xerox PARC

“Wire” Formats and Binding

Distributed object systems communicate with one another using messages. The term **wire format** describes both the set of messages defined, and the way in which the message is mapped to bits ‘on the wire’.

Binding is the way in which an object is described by a service so that a client can connect to it, and the process the client goes through to accomplish that connection

What's in the Wire Format?

Messages go back and forth between client and server.

Typically RPC-like protocols (OMG IIOP, DCOM) have **request** and **reply** messages.

Parameters are typically marshalled according to some form of **marshalled data representation** (MDR) such as OMG IIOP's CDR, DCOM's NDR, ONC RPC's XDR.

Additional messages can be added to support system features like **location forwarding** (OMG IIOP) or **distributed garbage collection** (ILU).

Elements of a Request Message

Basic:

- Request serial number
- Wire format version number
- Operation identifier
- Discriminant identifier
- Parameters

Also possible:

- Context information
- Caller identity information
- ‘Magic’ number
- Wire format negotiation information

Elements of a Reply Message

Basic:

- Request serial number
- Status indicator
- Return results

Also possible:

- Server identity information
- Wire format negotiation information
- Location forwarding information

Other Features of Wire Protocols

Concurrent method invocations allowed?

Type of transport (TCP/IP, shared memory, multicast, etc.)

Multiplexing of virtual connections over single transport connection?

Connection state at client end? At server end?

Communications security? Identity mechanisms? Encryption of messages?

Implementation complexity?

Marshalled Data Representations

Byte order: Big-Endian, Little-Endian, or receiver-makes-right

Alignment: ‘natural’, 4-byte, pad-before, pad-after

Mapping to C data structs (and re-arrangement of parameters)

Representation of:

- Strings (8-bit and Unicode)
- Objects
- Pointers
- Bulk Data

Additional Sub-Protocols

Location forwarding (OMG IIOP?)

Cache consistency (Java RMI?)

Distributed garbage collection (ILU, Modula-3 network objects)

Orderly shut-down (OMG IIOP, NEO)

Binding

Distributed object systems usually have some form of **static object reference** that can be passed around as data (OMG IIOP IOR's, DCOM string-form monikers, ILU string binding handles).

Static object references may be passed as data in a call, or may be obtained from some form of **binding service** or **name service**.

Features of Static Object References

Typically three elements:

- Object identifier (UUID or server-relative ID?)
- Object ‘type’ (operations supported)
- Communication information:
    - Wire protocols
    - TCP/IP endpoints
    - Multiplexing information

Also:

- Property list
- Security information

Binding Services

Simple (maps object UUID to static object reference)?

Complex (OMG Trading service)?

inetd-like (starts services or creates objects on demand)?

LAN-capable or WAN-capable?
If WAN, what federation protocol?

Internet possibilities: DNS? svrloc?