Remote Procedure Call

Peter Chapin
Vermont State University

Network Programming is Hard

- “Traditional” network programming is complicated.
  - Must deal with a special API
    - Create socket.
    - Create connection.
    - Format data into raw octets.
    - Write raw octets into connection.
    - Read raw octets from connection.
    - Interpret the raw octets.
    - Close connection.
    - Close socket.

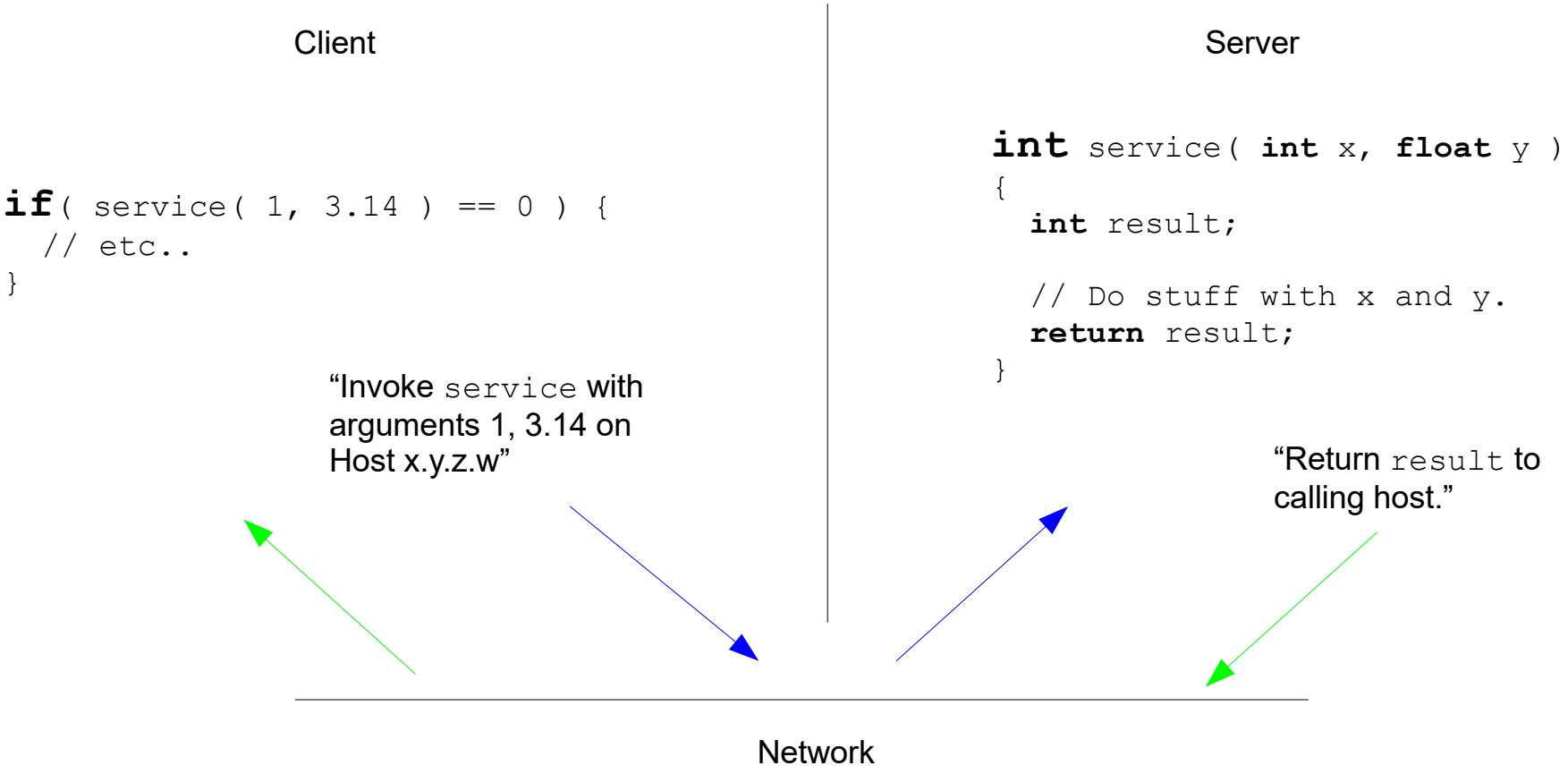
Streaming API is Easier

- Treat the network like a file.
  - Open connection.
  - Write various data types into the connection.
    - Integers, Floats, Characters, Strings.
    - More complex user defined types.
  - Read various data types from the connection.
    - Must distinguish one from the other somehow.
- Problem...
  - Application must still manage process and state.
    - What do the sent/received data objects mean?

Awkward Programming Model

- Program components...
  - Don't normally interact by passing raw bytes.
  - Don't normally interact through heterogeneous files.
- Instead...
  - Program components interact by *calling code* in separate libraries.
    - Data passed and returned as typed values.
- Can this be done over the network?

Remote Procedure Call (RPC)



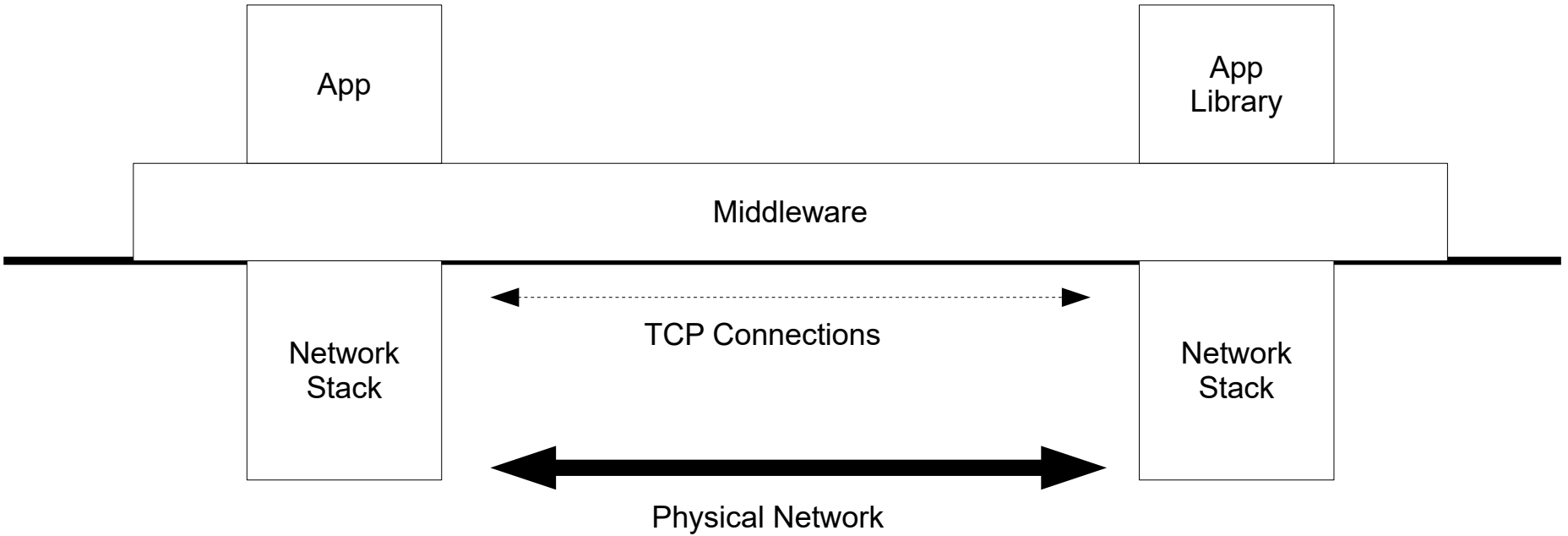
RPC

- Provides a natural programming model.
  - Makes network programs easier to understand.
    - ... easier to write, maintain, etc.
- Gives illusion of a single application.
  - Application is now distributed over the network.
  - *Distributed Application Programming*
  - Server appears as a library of functions.
    - Can be called in any order desired.
    - Data on the network is fully typed.

Disadvantages of RPC

- Requires software to...
  - *Marshal* (“encode” or “serialize”) and *unmarshal* (“decode” or “deserialize”) the parameters and return values.
    - Converts typed data to raw octets.
    - Must do so in a way that is understandable to peer.
  - Locate the peer, establish connection, etc, etc.
    - In short... take care of all the grunt work that was done by the programmer using the old way.
- This software is called *middleware*.
- Ideally...
  - The middleware would make the network “invisible.”

Distributed Applications



Middleware Technologies

- Many technologies to do this exist.
  - Sun's RPC (called ONC RPC)
    - Uses C
    - The basis for NFS and related technologies.
  - DCE (Distributed Computing Environment)
    - Based on old technology.
    - Now open source: http://www.opengroup.org/dce/
  - XML-RPC
    - Uses XML to structure data on the wire.
  - SOAP
    - Simple “Object” Access Protocol

More Technologies

- The list goes on...
  - Microsoft's history in this area is long.
    - COM became DCOM
    - MS embraced SOAP and web services.
    - .NET remoting.
    - “Windows Communication Foundation” (WCF)
  - If you are a Java person...
    - Java RMI (Remote Method Invocation)

Even More Technologies

- CORBA
  - Common Object Request Broker Architecture.
    - Standard controlled by the Object Management Group (OMG, see: http://www.omg.org)
    - Supports multiple languages.
    - Supports multiple platforms.
    - Supports multiple network protocols.
    - Large standard with many features.
      - Complex
      - Not fully implemented by anyone.
      - “Does everything.”
    - *Slowly vanishing.*

Ice

- Internet Communications Engine.
  - Created by two CORBA experts who became disillusioned by the CORBA “process.”
    - Supports multiple languages.
    - Supports multiple platforms.
    - Uses just TCP/IP.
  - Well designed, modern, open source.
    - Created originally to support a MMOG
      - The game failed, but not because of Ice.
    - A very nice system overall, but relatively unknown.
    - See http://www.zeroc.com/