

# **From RPC to Web services**

Eddie Aronovich

Eddiea[at]cs.tau.ac.il

# What's the need ?

Large complicated programs can:

- Do anything in one program
- Split tasks between multiple processes

# How far is remote ?

Processes can be in same:

- Computer
- Network
- (or) Any connected network / host

# Which services is it good for ?

- Resource sharing (RFC 707, Jan 76)
- Distributed processing (message passing)
- External software services (think of dll that runs on a remote system)

All those are not offered using regular I/O

# Historical evolution

- 80's— RPC (mainly SUN)
- 90's— Corba (ver 1.1-1991, 2.0-1996,  
3.0 - 2002)  
DCOM (1995)
- 00's- Soap (Open standard)

# Aren't sockets better ?

- Not for all tasks !
- The services that remote processing offers:
  - Service locator
  - Parameters passing
  - Process synchronization
  - APIs to different systems/programming languages
- Can handle many concurrent users at same time.

# RPC components

- Remote Procedure Calls – RFC1831 (ver. 2)
- Port Mapper
- XDR – eXternal Data Representation

# Some history

- White 76 (RFC 707)
- Xerox beginning of 1980s
- Sun releases first ONC<sup>(1)</sup>/RPC version (a.k.a SUN/RPC)

(1)Open Network Computing

# Port Mapper

- Each service access point is identified as a socket.

socket={ip-addr:socket}

- Client should know the socket the server waits on.
- Port mapper locates port of service on server given service id on computer.
- Port mapper allows multiple versions in same system.

# The X file...

```
struct square_in { /*Input Argument */
    long arg1;};

struct square_out { /*Output Argument */
    long res1;};

program SQUARE_PROG {
    version SQUARE_VERS{
        square_out SQUAREPROC(square_in)=1;
    } = 1;
} = 0x3123;
```

# Server (works hard 😊)

```
#include "square.h"
#include <math.h>

square_out * squareproc_1_svc(square_in *inp,
    struct svc_req *rqstp) {

    static square_out out;
    out.res1 = sqrt(inp->arg1);

    return(&out); }
```

# Client program

```
#include "square.h" /* generated by rpcgen */
int main(int argc, char **argv) {

    CLIENT *cl;
    square_in in;
    square_out *outp;

    if (argc != 3)
        err_quit("usage: client <hostname> <integer-
value>");
    cl = Clnt_create(atoi(argv[1]), SQUARE_PROG,
        SQUARE_VERS, "tcp"); in.arg1 =
    atol(argv[2]); if ( (outp = squareproc_1(&in,
cl)) == NULL)
        err_quit("%s", clnt_serror(cl, argv[1]));
    printf("result: %ld\n", outp->res1);
    exit(0); }
```

# Why RPC is not enough

- RPC assumes the processes are from same user (in same accounting system).
- It is limited to very well defined tasks.

# CORBA

## Common Object Request Broker Architecture

- Object resides on system A invokes method on system B.
- The programmer does not know where & how an object is implemented.
- IDL<sup>(1)</sup> and ORB<sup>(2)</sup> are used to describe & communicate between processes
- First security services were introduced in 96(ver 2.0)!

# Corba vs. RPC

- Using RPC data type of parameters is fixed unlike corba which is more flexible.
- RPC Language partially dependent, corba is language independent
- RPC does not support push operation, while CORBA does.
- There was no real interoperability ORB. Each vendor tried to dominate both client & server side

# Web services

- Open standard architecture
- Allows using entities from remote machines
- Have security and environment preparations for the World Wild Web

# What are the components

- SOAP – Simple Object Access Protocol
  - What is in the message
  - Who should read it
  - Optional / Mandatory
- WSDL - WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information.

# Where is my service ?

- UDDI provides three basic functions:
  - Publish: How the provider of a Web service registers itself.
  - Find: How an application finds a particular Web service.
  - Bind: How an application connects to, and interacts with, a Web service after it's been found.