

# Open Floppy Grid

Joe Armstrong

[joe.armstrong@ericsson.com](mailto:joe.armstrong@ericsson.com)

Programming distributed systems  
with  
XML-RPC Corba SOAP  
WSDL J2EE C++ Sockets  
is

AMAZINGLY  
DIFFICULT

it

should

be

Really

Easy

# Why is this?

XML - RPC - flawed cannot specify an RPC

Xml based things - flawed XML sucks

Corba - amazingly difficult

SOAP - WSDL - sucketh greatly

All these things are very  
difficult to implement

# How can we connect things together?

Pipes?

A | B | C

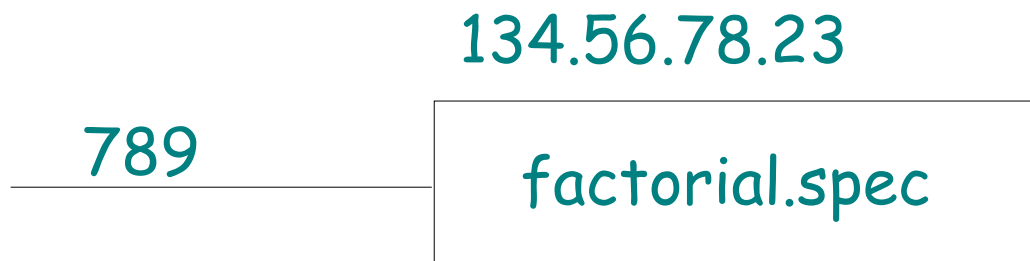
Very very very easy - but

- Not distributed
- Need ad-hock parsers
- No specified grammars

# Contracts

Bind: 134.56.78.23 789

<http://www.sics.se/~joe/contracts/factorial.spec>



`{"factorial", int()} => int();`

A red arrow points from the URL 'http://www.sics.se/~joe/contracts/factorial.spec' to the contract definition box.

# The contract

`{"factorial", int()} => int();`

Means that this

Send:

Content-Type: text/ezXML

Content-length:NNN

`<t><s>factorial</s><i>123</i></t>`

Receive:

Content-Type: text/ezXML

Content-length:NNN

`<i>175177157173500</i>`

is a legal interaction

# Types

# Encodings

```
int()  
int(Min..Max)  
str()  
{T1, T2, T2, ...}  
[T]  
T1 | T2 | T3
```

```
<i>NNNN</i>  
  
<s>SSSSSSSS</s>  
  
<t> E(T1) E(T2) ... E(Tn) </t>  
  
<l> E(T1) E(T2) ... E(Tn)</l>
```

Note: must quote "<" within <s>..</s>

# Contract Syntax

```
newtype() = newtype() = ... = Type;  
Type => Type;           // RPC  
Type <= Type;          // Reverse RPC  
empty() => Type;        // notification  
Type => empty();        // event
```

This is A DTD (schema) and a protocol specification

$A \Rightarrow B$  means "if you send me a message of type A I promise to return a message of type B"

# file\_store.spec

```
dir() = fileName() = file() = fileContents() = str();  
"ls" => [ {"dir", dir()} | {"file", fileName()} ];  
{"get", file()} => {"ok", fileContents()} | "noSuchFile";  
{"put", file(), fileContents()} => "ok" | {"error", str()};  
"pwd" => dir()  
{"cd", dir()} => {"ok", dir()} | "no"
```

# meta.spec

```
protocolName() = ip() = language() = str();
port() = sessionId() = int();
"vsN" => {"vsN", int()};
"listProto" => {"ido", [protocolName()]};
{"become", protocolName()} => {"iamnow", protocolName()};
"describe" => {"iamnow", protocolName()};
empty() => {"jumpTo", ip(), port(), sessionId()};
// (resume session on ip())
{"talkToMeIn", [language()]} => ({"yes", language()} | "no");
{"whoAreYourFriends", [protocolName()]} =>
    {"myFriendsFor", protocolName(), [ip()]};
{"iAmYourFriend", protocolName()} => 'empty';
```

<http://www.sics.se/~joe/contracts/>