CORBA & Interpreted Languages

Duncan Temple Lang

August 10, 1998

`cm.bell-labs.com/stat/duncan/DistComp`

Lucent Technologies
Bell Labs Innovations



Outline

- CORBA:
  what is it?
  how do we program with it?
- CORBA & Interactive Languages.
- Support Tools.
- Outstanding Issues.
- Alternative Approach – Ω.

CORBA

- General remote method invocation mechanism
  CGI, PVM/MPI, DCOM and Java's RMI.
- Client-Server style.
- Object based and type checked.
- Language independent.
- Machine independent - location and architecture.
- Myriad of additional services:
  Naming, Event, Security, Transaction, . . .

Programming With CORBA

- Describe Service

  IDL - Interface Definition Language.

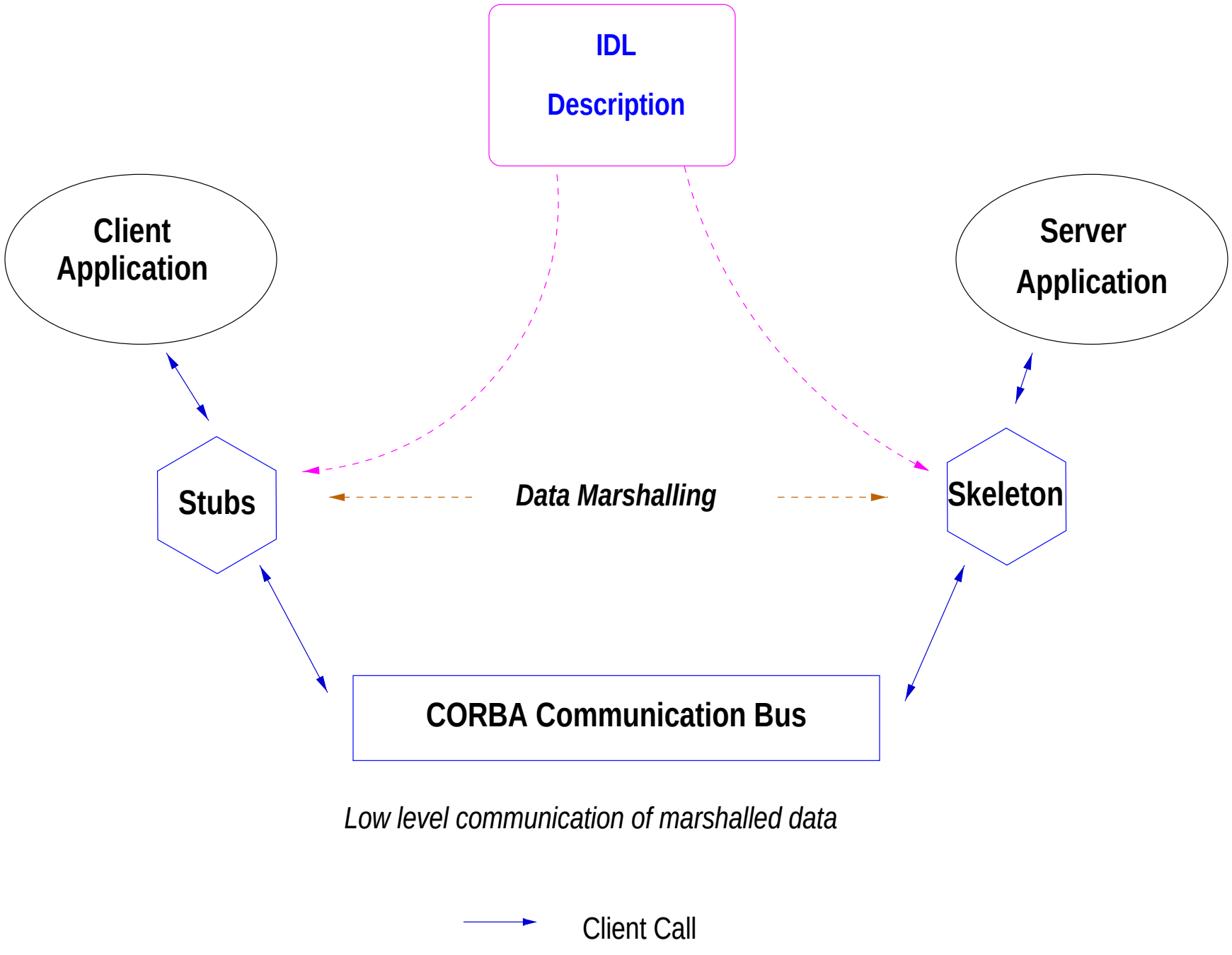
```
interface PlotServer {
 boolean histogram(DoubleArray x);
 boolean plot(DoubleArray x,
              DoubleArray y);

 Options options();
};
```

Programming With CORBA, ctd.

Client-side

- Compile IDL stubs for chosen language
  e.g Java.
- Connect to ORB.
- Locate server object.
- Treat server as if local object & invoke methods.
- “Compile” and run!


IDL
Description
Client
Application
Server
Application
Stubs
Data Marshalling
Skeleton
CORBA Communication Bus
Low level communication of marshalled data
Client Call

Programming With CORBA, ctd.

Server-side

- Write IDL description.
- Compile IDL skeleton for chosen language e.g. *C++*.
- Extend skeleton methods with actual code.
- Add code to
  - connect to ORB,
  - identify server to CORBA bus.
- “Compile” application and run!

Interactive Languages

- No compilers available for high-level statistical languages.
- Solution: Develop compiler(s), or
- Replace all Stubs and Skeletons with single generic mechanism.
- Requires Reflectance/Self Description.
  Ability to query language about itself.
  Methods, classes, etc.
  Only for server.
- Perform operations using CORBA's own reflectance &
  Dynamic Invocation Interface (DII) &
  Dynamic Skeleton Interface (DSI).

Dynamic Client Invocations

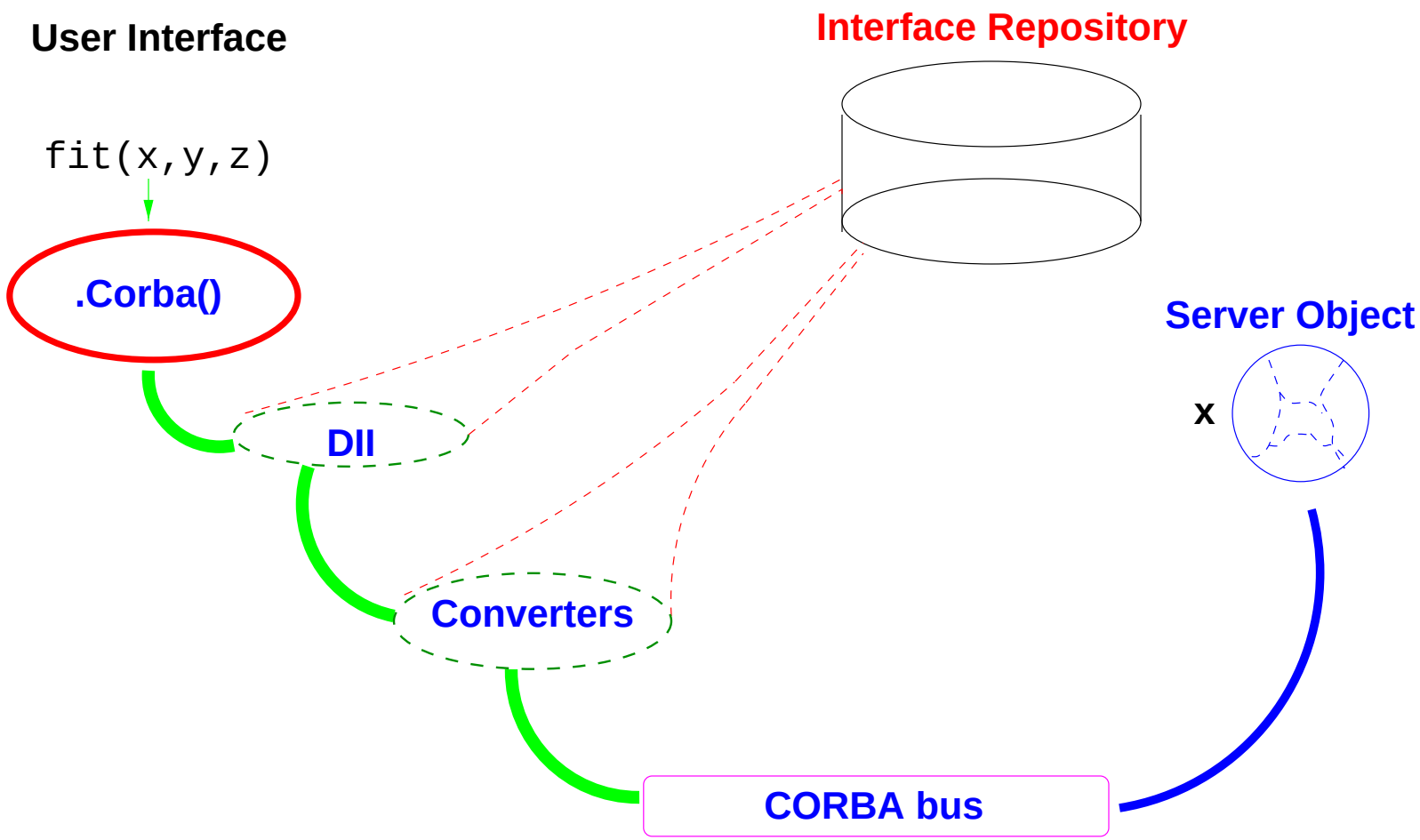
- User Interface
  - S4

    ```
    .Corba(server, "plot", x, y)
    ```

  - Java

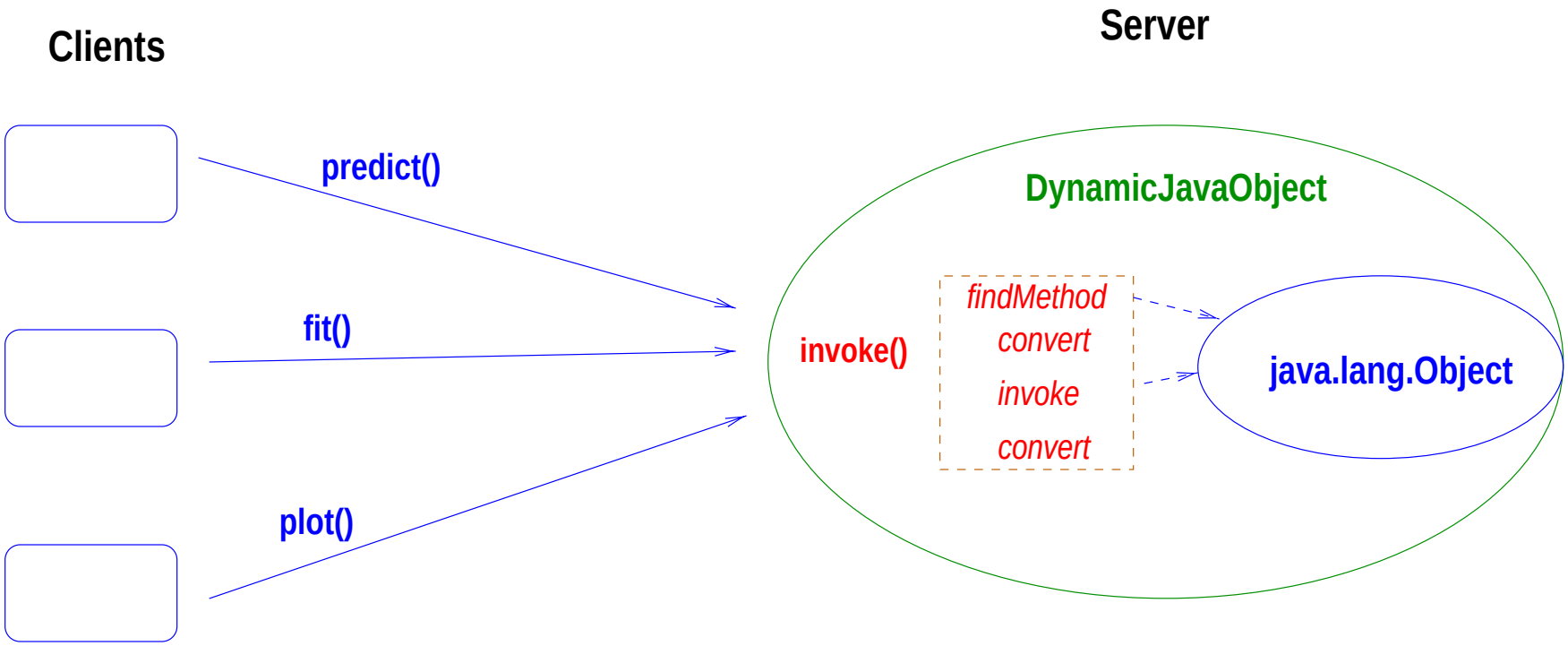
    ```
    args[0] = x; args[1] = y;
    new CorbaCall(server,"plot",args);
    ```

- “Algorithm”
  - Convert arguments to types described in IDL operation description.
  - Construct request contents - return type and arguments.
  - Invoke request.
  - Convert CORBA result to native object.
- Allows asynchronous and one-way invocation.


User Interface
Interface Repository
fit(x,y,z)
.Corba()
Server Object
x
DII
Converters
CORBA bus

Dynamic Server

- Makes CORBA server from native object.
- Inputs: object and IDL description name.
- All requests received by `invoke()`.
- Algorithm
  - Convert arguments to native types.
  - Search for method using operation name & arguments.
  - Evaluate method.
  - Convert result to appropriate CORBA object.


Clients
Server
predict()
fit()
plot()
DynamicJavaObject
invoke()
findMethod
convert
invoke
convert
java.lang.Object

Comments

- Generic algorithms.
  - No direct knowledge of IDL for server
  - Use only CORBA functionality.
- Converters are language specific.
- Extensibility for user–level converters.

Issues

- Dynamic instantiation of unseen classes.
- Understanding servers' methods - dynamic documentation.
- Efficiency.
- Configuration of distributed environment.

Support Tools

- IDL generators.
  - From Java and S4 classes.
  - Allows polymorphism using DSI.
- Naming Service access -
  available servers by name.
- Interface Repository queries -
  descriptions of available modules, interfaces, operations, etc.
- Future:
  - CORBA Events
  - High Performance Computing

An Alternative Approach - Ω

- S, Matlab – interpreters *on top of* C/C++.
- Interpreter in Java
  - Parser
  - Reflectance
- Access to existing Java classes
  - Databases
  - GUIs
  - network - URLs, http, sockets, etc.
- Ability to compile.
- Ω project
  R: Robert Gentleman, Ross Ihaka
  S: John Chambers, Duncan Temple Lang
  Xlisp-Stat: Luke Tierney
- Will rely on broad participation.