

Prototype of a Simple ZeroMQ-Based RPC in Replacement of CORBA

Y. Le Goc, F. Cecillon, C. Cocho, A. Elaazzouzi, J. Locatelli, P. Mutti, H. Ortiz, J. Ratel

Abstract The NOMAD instrument control software of the Institut Laue-Langevin is a client server application. The communication between the server and its clients is performed with CORBA, which has now major drawbacks like the lack of support and a slow or non-existing evolution. We present the prototype of a simple RPC built on top of ZeroMQ and the performant Google Protocol Buffers serialization tool, to which we add a remote method dispatch layer. The final project provides a CORBA IDL compiler restricted to a subset of the language so that only minor modifications to our existing IDL interfaces and class implementations will have to be made to replace the communication layer in NOMAD.

How to Replace CORBA?

- Many existing CORBA IDL files
- Need for message reception guarantee

Solution

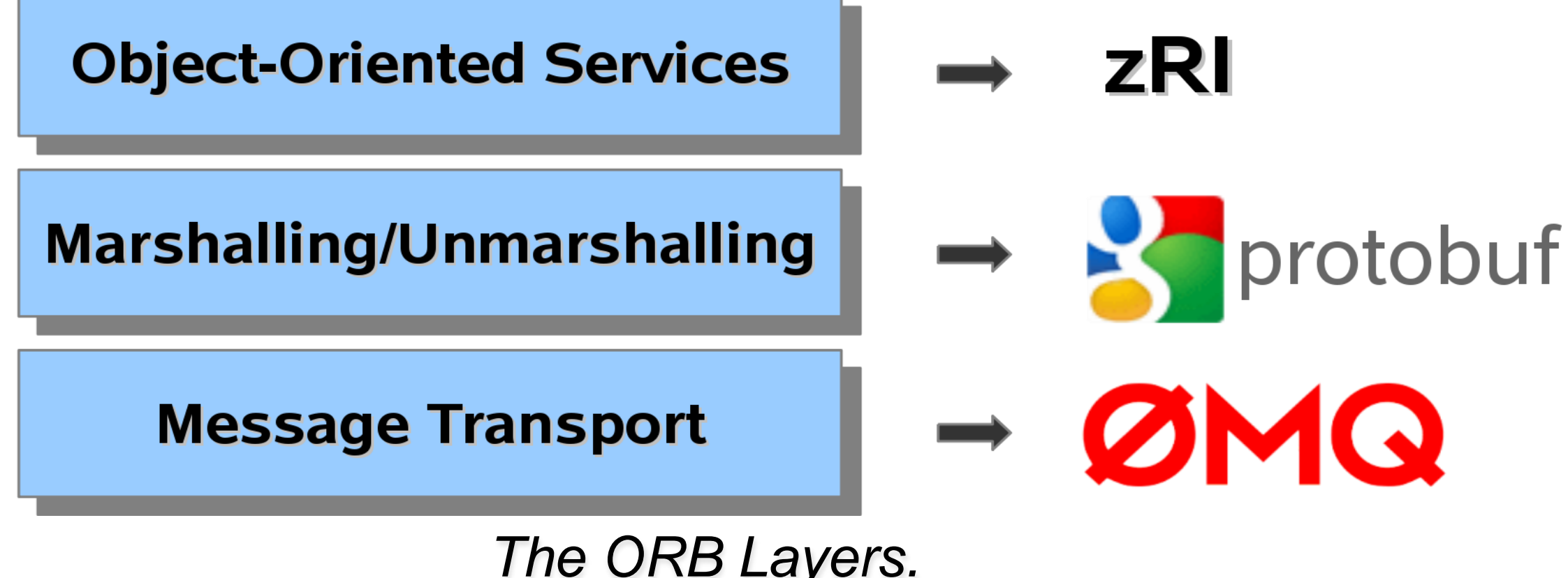
- ZeroMQ + Protocol Buffers marshalling
- Implement a simple *Object Request Broker*

Features

- CORBA IDL language subset
- Remote method invocation layer
- Only arguments by value
 - No complex interactions between client and server
 - No object reference passed
- Automatic serialization of arguments and return values
- Synchronous client and server calls

Project zRI

- zRIg, an IDL compiler
- zRIcpp, C++ runtime
- zRIj, Java runtime



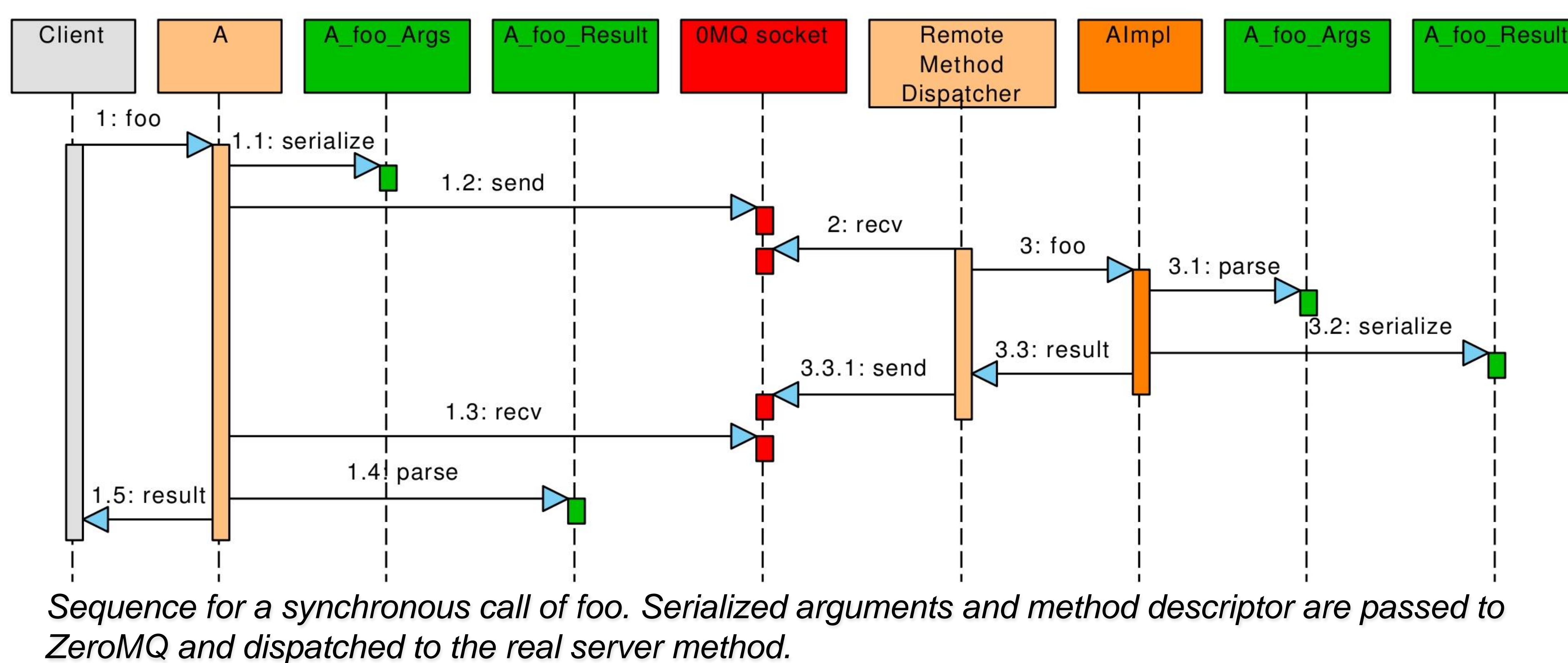
Advantages

- Reuse the existing IDL files + implementations
- Hides Protocol Buffer layer
- Object integration easier
- Type-checking at compile time

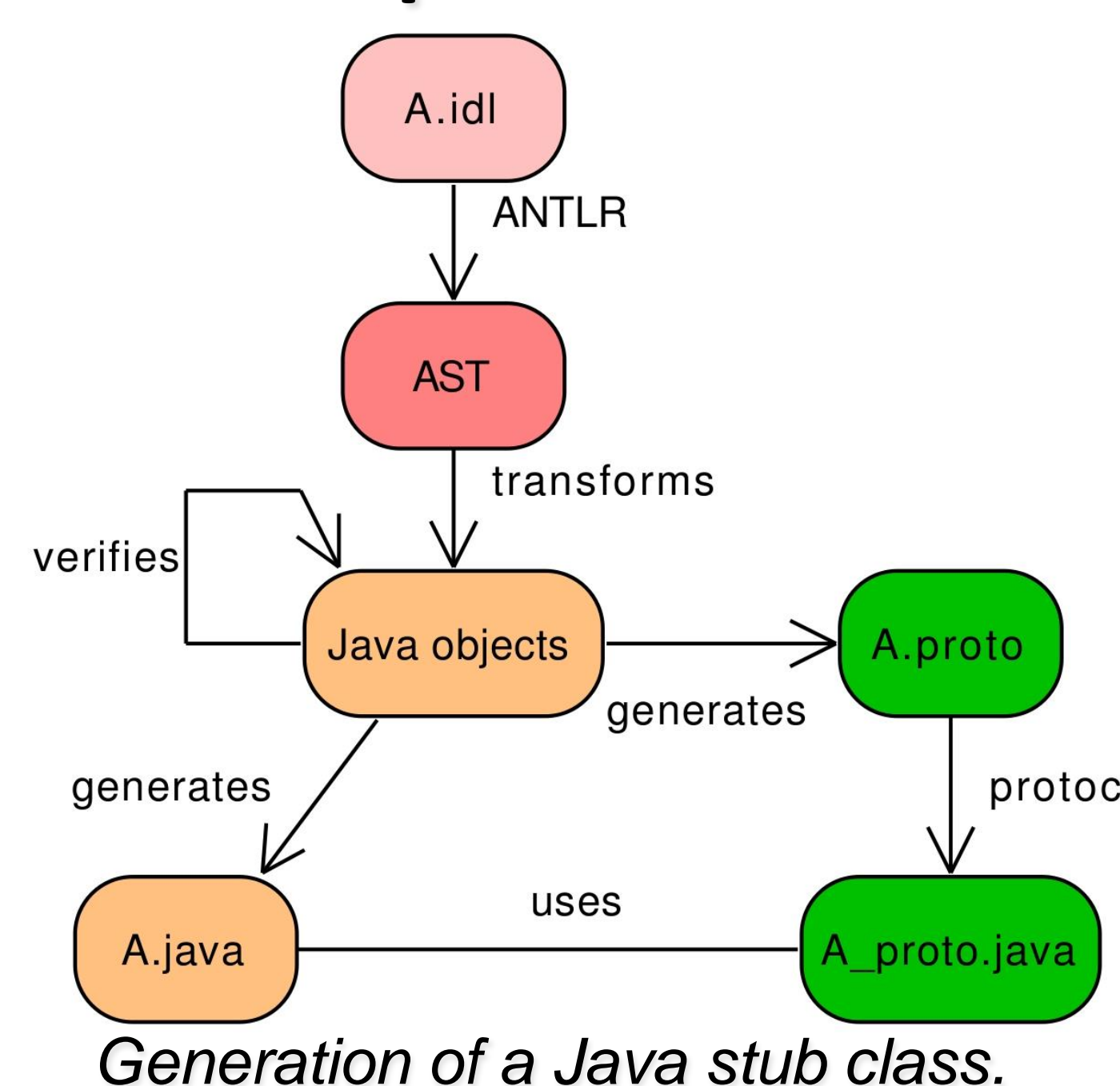
Execution Example

- Basic A.idl file


```
interface A {
    double foo(in short a, in boolean b);
};
```
- Client side
 - Serialization of arguments
 - Remote method call
 - Deserialization of return value
- Server side
 - Deserialization of arguments
 - Real method call
 - Serialization of return value
- Synchronous call by socket REQ/REP



IDL Compiler



- Java application
- ANTLR 3.5-based
- Generates
 - Temporary *proto* file
 - Serialization Protobuf classes
 - Stub and Skeleton classes
- CORBA IDL subset
- Easy to extend

Performance Comparison with CORBA

- Test on large arrays
 - omniORB 4.1.4 + JacORB 2.3.1
 - Intel Xeon 2.40GHz, 4GB, Suse 11
- Results
 - CORBA 2x faster than zRI
 - Protobuf unmarshalling bottleneck
 - Further optimizations should provide equivalent results

