A Brief History of Distributed Programming: RPC

Code Mesh London 2016
Caitie McCaffrey
Distributed Systems Engineer

@caitie
caitiem.com
Concurrent Programming

Computation can advance without waiting for all other computations to complete

Distributed Programming

Computation is accomplished via the communication and coordination of networked computers
“An asynchronous event driven JavaScript runtime”

Single Threaded
Event Loop
Callbacks
Go Routines

Channels
How do you communicate with code running on other machines?
How do you communicate with code running on other machines?

We’ve left that as an exercise for the reader...
Actor Model for Concurrency & Transparent Distribution Added Later
Cloud Haskell

Haskell Implementations of Distributed Erlang semantics

Extension of a Concurrent Language for Distribution
RPC: Remote Procedure Call

make remote calls just as simple as local calls
Procedure Call Paradigm (PCP) is an attempt at defining a mechanism for resources sharing across all 70 nodes on the Internet.
RFC 684 Criticisms

Local Calls & Remote Calls have different Cost Profiles

Remote Calls can be delayed or never return

Asynchronous Message Passing is a better model because it makes the passing of messages explicit
Generalization to functions

Generalize TELNET and FTP’s call-and-response model to functions from an application-specific grammar. One port for all protocols.

Control Flow Critique

RPC only allows for sequential composition.
Implementing Remote Procedure Calls

First Commercial Grade RPC

Fig. 3. The packets transmitted during a simple call.
We propose the following test for a general-purpose RPC system
“Imagine that two programmers are working on a project. Programmer 1 is writing the main program. Programmer 2 is writing a collection of procedures to be called by the main program”
“At the very last minute, after all the code has been thoroughly tested, debugged, and documented and both programmers have quit their jobs and left the country ...”

A Critique of the Remote Procedure Call Paradigm
“...the project management is forced by unexpected, external circumstances to run the program on a distributed system.”

A Critique of the Remote Procedure Call Paradigm
“...the project management is forced by unexpected, external circumstances to run the program on a distributed system.”

A Critique of the Remote Procedure Call Paradigm
“It is our contention that a large number of things may now go wrong due to the fact that RPC tries to make remote procedure calls look exactly like local ones, but is unable to do it perfectly”

A Critique of the Remote Procedure Call Paradigm
“There is, in fact, no protocol that guarantees that both sides definitely and unambiguously know that the RPC is over in the face of a lossy network.”

A Critique of the Remote Procedure Call Paradigm
RFC 1094

Network File System

First major distributed filesystem that gained popularity and adhered to the existing UNIX filesystem API
Network File System

Soft Mounting  Introduced new error codes for distributed failures that existing UNIX applications could not handle

Hard Mounting  Operations would block until they could be completed successfully
CORBA
Common Object Request Broker Architecture

Supported Cross-Language, Cross Address Space Interoperability for Object-Oriented Programming

Interface Definition Language (IDL): used to generate stubs for remote objects & mappings between different primitive types

“It’s Just a Mapping Problem” Remote to local exception mapping, remote to local method invocation.
It is the thesis of this note that this unified view of objects is mistaken.
A Note on Distributed Computing

Latency

Performance analysis is non-trivial and one design is not always going to be the right design.

Memory Access

How do we deal with the problems of pointers and references? Once moved they are no longer valid unless we use distributed shared memory.

Partial Failure

Failures are detectable in the local environment and result in a “return of control”. In distributed computing this isn't true.
Two Paths Forward

Treat all objects as local

Or

Treat all objects as remote

A Note on Distributed Computing
Treat all objects as remote

“This approach would also defeat the overall purpose of unifying the object models. The real reason for attempting such a unification is to make distributed computing more like local computing and thus make distributed computing easier. This second approach to unifying the models makes local computing as complex as distributed computing.”

A Note on Distributed Computing
Present
Microservices
Microservices
The Re-emergence of RPC Frameworks
Finagle

RPC System for the JVM

IDL: Thrift

Based on Futures

Request/Response
gRPC

IDL: Protobufs

Multi-Language RPC framework
(C/C++, C#, Node.js, PHP, Ruby, Python, Go, Java)

Supports Bi Directional Streaming
Modern RPC Frameworks

Don’t Provide a Unified Model
“The hard problems in distributed computing are not the problems of getting things on and off the wire.”

—A Note on Distributed Computing
The point of RPC was to make remote calls just as simple as local calls.
The point of RPC was to make remote calls just as simple as local calls.

If we treat everything as remote, have we simplified distributed computation at all?
The point of RPC was to make remote calls just as simple as local calls.

If we treat everything as remote, have we simplified distributed computation at all?

If we can't treat all calls as local, is the *procedure call* the right abstraction for distributed computation?
Future

Spores: A Type-Based Foundation for Closures in the Age of Concurrency and Distribution

Lasp: Distributed Deterministic Data Flow Programming for Erlang

Consistency Analysis in Bloom: a CALM and Collected Approach
Spores are small units of possibly mobile functional behavior.

Serializable closures with capture controlled by the type system (Scala).

Dual to Actor Systems (like Erlang). Actors exchange data with async messaging, spores are stateless processes that pass functions around with asynchronous messages.
Thanks

Resources
https://github.com/CaitieM20/Talks/tree/master/DistributedProgrammingRPC