Convenience Over Correctness

Sudeep Reddy Eleti

Agenda

This lecture provides an overview of

- Programming languages for distributed systems and RPC influence on them
- Fundamental flaws in RPC
- Developer convenience over RPC flaws
- Alternatives

Introduction

- **Distributed System**

  It is a software system in which components located on the networked computers communicate their actions by passing messages.

- **The distributed systems were first programmed using the sequential programming language**

Influence of RPC

- Distributed Systems like Argus and Emerald, first explored the probabilities for programming languages to be distributed.
- Later in 1980's, Apollo NCS, Sun RPC provided full distributed computing capabilities using general purpose languages like C and Pascal.
- In 1990's, the Corba and Microsoft COM developers used distributed objects.

Downfall

- Developers built many applications using this technologies.
- But, most of the applications are gone now and the newer ones are waning.
- For example, Corba systems are still there, but most treat them as legacy technologies.

RPC Flaws

- The reason is RPC is fundamentally flawed
- RPC is programmed in similar way as Local Procedure Call.
- However, the Remote and Local invocations have different characteristics with respect to latency, memory access, concurrency and partial failures.
- Still, the researchers are trying to develop new RPC oriented systems. Why?

Convenience of Developer

- RPC oriented systems provide familiar programming language constructs to remote services.
- In Java, the remote services are represented as objects and method functions.
- In C, the remote services are represented as functions.
- The calling site and receiving site in a remote procedure looks no different from ordinary local method or function call.
- So, developer is in comfortable confines of their programming languages. But, is developer convenience more important than other concerns of RPC?

Other Problems with RPC

- RPC systems use the IDL's to map the data types from one programming language to other language. However, this mapping is imperfect.
- Distributed Systems require intermediaries to ensure that system will operate and perform as required. But RPC doesn't deal with it.
- RPC doesn't talk about caching which is critical for scaling.

Alternatives

- **Message Queuing Systems**

  The developers build their applications such that their code directly deals with the network and its effects.

- **REST (Representational State Transfer)**

  It deals with caching and intermediaries.

Contd…

- So why were these systems not used?

  None of the message queue systems were freely available till recently.
- Now the scenario is changing and its time for RPC to retire.

Summary

- Flaws in RPC are more important than developer convenience and has to be dealt to develop good distributed systems.
- Message queue systems and REST as alternatives for RPC.

Thank You