Developping distributed systems with the actor model

Jonathan

December 25, 2018

Contents

1 Introduction

Developping distributed systems is not a simple task as it involves dealing with problems such as concurrency, fault tolerance, and scaling. Over the last decades, several models has been proposed to organize computing resources working in an distributed systems. One of them is the actor model proposed by Carl Hewitt in 1973 [**?**], which has inspired a language such as Erlang, or frameworks for distributed systems such as Akka.

Roughly, the actor models proposes to organise distributed systems according to the following principles:

1. Everything in the system is an actor
2. An actor can receive and send messages to other actor (including itself)
3. An actor can process one message at a time
4. at the end of 3., an actor can send one or several messages to one or several actors

5. at the end of 3., an actor can changes its behaviour for the next message

6. at the end of 3., an actor can create new actors

Actors have private states that are not directly shared with other actors. Instead, they collaborate via exchanges of messages. The lack of shared states prevent the need for synchronization systems such as lock or mutex. Thus, the actor model is a good fit for building distributed systems or highly concurrent applications.

2 Objective of the session

The objective of this session is double : first we will develop a simple client/server application using Akka. Second, we will reuse the code developed for the first step and implement the Chord algorithm [**?**], a well know fully decentralized algorithm.

3 Using Akka to develop distributed software

4 Implementing Chord