Advanced Programming

Assignment 3 - `Mayo Peer-to-Peer`

Kasper Nybo Hansen and Mads Ohm Larsen

October 11, 2010

1 Introduction

This week we had to create a concurrent peer-to-peer (P2P) phonebook program in Erlang. The implementation should be an API that allows the following operations:

- Add a contact
- List all contacts
- Update a contact, given the name
- Delete a contact, given the name
- Find a contact, given the name

The individual peers should also be able to join the network and share the contacts.

2 The Mayo Routing Algorithm

We are using the Mayo Routing Algorithm to maintain our contacts in our phonebook. Each peer have got a GUID, which is a 128-bit MD5-hash of their name. Each peer have got the contacts which has a higher MD5-hash than the peer, up til the next peer. The left most peer have also got the contacts below.

When doing this there were some thing we had to take into mind. What happens if P_1 and P_2 makes a network together, and shares some contacts, and P_3 and P_4 makes a networks and shares some contacts, and P_3 joins P_1? In our implementation this results in the P_3/P_4 network all call the `join`-method on P_1, and are routed into the correct place. The contacts are also all given to P_1, which finds the correct peer for the contact, and gives it to that peer (Might be P_3 or P_4, but could equally likely be P_2).