UNIVERSITY OF THESSALY

Department of Computer & Communication Engineering

Mobile/PC mixed environment support for POBICOS middleware

Petros Kalos
petrkalos@gmail.com

https://github.com/petrkalos/pobicos2pc

Supervisor: Spyros Lalis

Introduction

Goals

Porting POBICOS middleware for PC therefore we set up an hybrid environment which contains mobile clients and pc (servers).

Servers and mobiles communicated directly (no through Directory)

Create a WebUI which can start/stop middleware and POBICOS application, print middleware and POBICOS application log. Also appear in a map the clients of the server

Installation Instructions

Prerequisites

Java Runtime Environment & JDK
C compiler (for compiling the middleware library)
Apache Tomcat and Vaadin Framework(for the WebUI)

Installation for Linux

Compile the middleware.c (middleware/middleware.c)
Compile the PoClientPC java project (set the path for the middleware library)
Compile the DirectoryService in order to have communication between nodes

First Run in local computer (for testing)

Run DirectoryService (java -jar DirectoryService.jar)
Run PoClientPC
If you want to load a POBICOS application you must run
    java -jar PoClientPC.jar “applicationpath”
If you want to start only the middleware run without “applicationpath”

pobicos2android[2]

Devices and Communication

The only devices in this project is mobile nodes with Android OS.

The mobile devices communicate with each other through a Directory & Forwarder Service with a TCP which is initiated by the mobile when it (re)connects to the Internet, and remains alive as long as the mobile remains connected.

Directory Service assign a virtual address to each device.

Forwarder Service get messages from devices and if the destination is currently connected to the forwarder, the message is propagated to it via the TCP connection that was opened by the mobile. Else, if the destination is not currently connected to the forwarder, the message is dropped (leaving it up to the POBICOS protocols to handle timeouts and retransmissions).

Why indirect communication?

- Due to NAT restrictions we cannot have direct communication between Android devices.
- Because devices doesn’t have stable internet connection (GPRS)

- **pobicos2pc**

  - **Devices and Communication**

    - Mobile Devices and stationary PC with stable Internet connection.

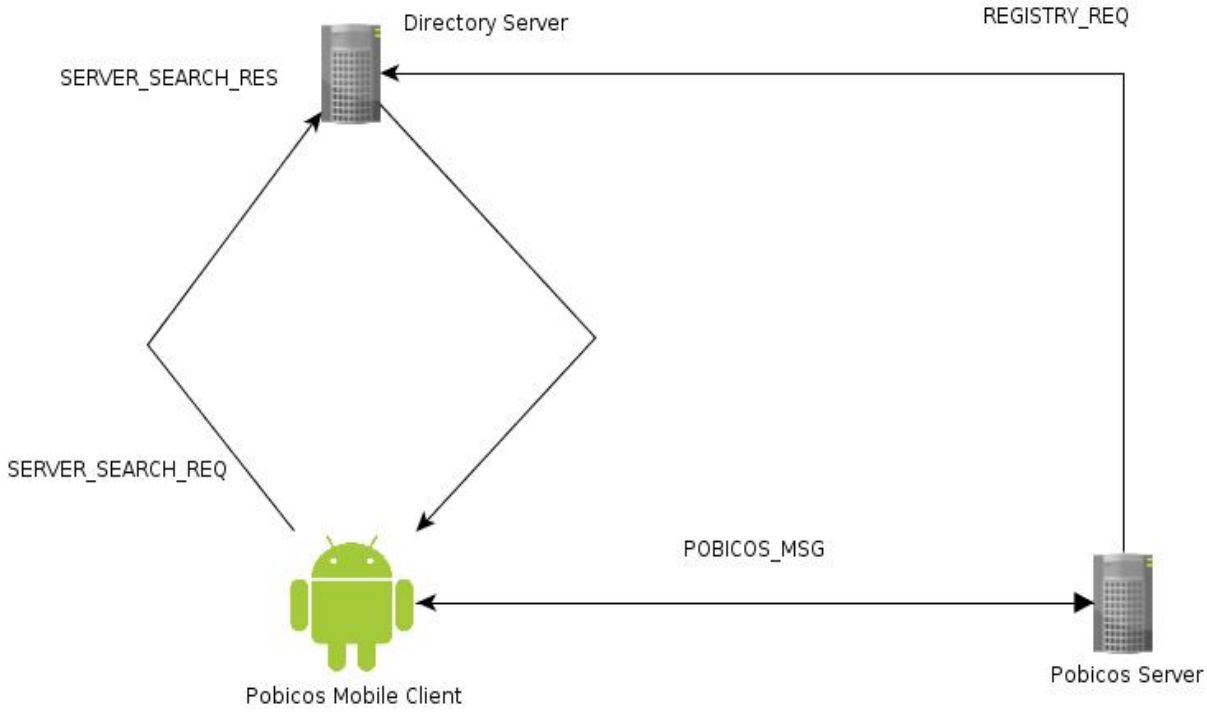
    - PC can now act like servers and accept direct messages from mobile and servers devices slide over the Directory and Forwarder.

    - Each host indicates via its registration whether it is a PC/server or a mobile, and the registry assigns special virtual addresses to PCs (e.g., most significant bit set). Also, the registry-host protocol is augmented by adding a request-reply transaction whereby a host can request/receive the IP address for a virtual address.

    - When a mobile wishes to send a message to a remote host, it inspects its address. If the destination is mobile, it sends the message to the forwarder, as pobicos2android. Else, if the destination is a PC, it retrieves the IP address from the registry (stores it locally in a cache for future reference) opens a TCP connection to the PC, and uses this connection to send/receive messages to/from the PC. If the connection stays idle for a long time, it is closed.

    - When a PC wishes to send a message to a remote host, it inspects its address. If the destination is mobile, and this has not yet established a TCP connection with the PC, the message is sent to the forwarder, as pobicos2android. Else, if there is a TCP connection with the mobile, the message is sent directly to it. Else, if the destination is a PC, it retrieves the IP address from the registry, opens a TCP connection to the PC and sends the message to it directly (each PC-pair may negotiate to share a single TCP connection for sending/receiving messages).

    - If direct communication of a host with a PC fails, the TCP connection is closed and the locally kept IP address information is invalidated (again, leaving it up to the POBICOS protocols to handle timeouts and retransmissions, which in turn will trigger a renewed attempt to communicate with the PC).


Directory Server
REGISTRY_REQ
SERVER_SEARCH_RES
SERVER_SEARCH_REQ
POBICOS_MSG
Pobicos Mobile Client
Pobicos Server

- ## Messages Form

  - General Message Format

NetworkMsg	MsgType PayLen Payload
MsgType	uint8_t
Paylen	uint8_t
Payload	{byte}*Paylen

  - ### Server Search Request

NetworkMsg	MsgType Paylen Payload
MsgType	SERVER_SEARCH_REQ
Paylen	uint8_t
Payload	ClassOfDevice ServerAddress
ClassOfDevice	CLASS_SERVER
ServerAddress	uint8_t

  - ### Server Search Response

NetworkMsg	MsgType Paylen Payload
MsgType	SERVER_SEARCH_REQ
Paylen	uint8_t
Payload	ServerIpAddress ServerPort PoAddress
ServerIpAddress	uint8_t*{ServerIpAdressLength}
ServerPort	uint32_t
PoAddress	uint16_t

References

1. pobicos project http://www.ict-pobicos.eu/
2. pobicos2android https://github.com/kl86/pobicos2Android
3. Vaadin Framework https://vaadin.com/
4. Apache Tomcat http://tomcat.apache.org/