Android GPS

Thomas Tallentire

Marc Rafanin

Marc Vouve

Lewis Scott

State Transition Diagrams

# Server STD



# Client STD

## 

*\*Client state diagram above is based on application activities.*

Client Class Diagram



Sample Data XML

The server will need the following XML data format from the client:

*<?xml version="1.0" encoding="UTF-8"?>*

*<DATA>*

*<ENTRY>*

*<TIME>10:44:56 PM</TIME>*

*<GPS>*

*<LATITUDE>49.266667</LATITUDE>*

*<LONGITUDE>-122.966667</LONGITUDE>*

*<ALTITUDE>30.54678</ALTITUDE>*

*<SPEED>2.44435</SPEED>*

*<HEADING>12.443665</HEADING>*

*</GPS>*

*<IDENT>*

*<IP>127.0.0.1</IP>*

*<HOSTNAME>localhost</HOSTNAME>*

*<MAC>000c299cb333</MAC>*

*</IDENT>*

*<INFO>*

*<IMEI>356938035643809</IMEI>*

*<DEVID>20013fea6bcc820c</DEVID>*

*<PHONE>6045553312</PHONE>*

*<GOOGLE>stalking\_victim@gmail.com</GOOGLE>*

*<ICON></ICON>*

*</INFO>*

*</ENTRY>*

*</DATA>*

Pseudocode

# Server:

### main

{

setup

loop

}

### setup

{

try file

print loading message

open socket

set reuse addr on socket

bind to socket

listen on socket

print ready message

}

### try file

{

open file

if file is invalid

print error message

quit

}

### loop

{

accept

print new client message

fork

}

### accept

{

accept from socket(blocking)

save new socket

}

### fork

{

call system fork

if in parent

parent

else

child

}

### parent

{

close client socket

loop

}

### child

{

request suicide on parent death from kernel

child loop

}

### child loop

{

while not done reading

read into buffer

jump back in file to before last DATA closing tag

write buffer to file

write closing DATA tag to file

child loop

}

# Client:

The pseudo code below is a pseudo code for Andriod activities/services. Refer to the detailed client pseudo code for class implementations.

### Home

{

get input from user for server ip and port number

launch sending screen with user input when go button is clicked

}

### Sending

{

when send button click or gps location changes

send gps update to server by calling SendData

and pass server ip and port

}

### SendData

{

connect to server

send gps data to server

}

# Detailed Client Pseudo Code:

Activity, service, and helper details will be listed in this section.

## HOME ACTIVITY

Initial activity when application launches

### onCreate

{

display default server host and port

display "go" button

}

### valid (ip, port)

{

call addressvalidation to validate ip

validate port

return true if valid ip and port

}

### AddressValidation

{

validate ip

}

### go

{

When go button clicked

launch Sending activity passing ip and port

}

## SENDING ACTIVITY

Activity that listens for gps location changes and initiates data sending to server.

### onCreate

{

get ip and port details from home activity

initialize gpsHelper and XMLHandler

}

### onResume

{

register receiver from gpshelper to get gps location changes

}

### onPause

{

unregister receiver from gpshelper

}

### sendLocation

{

get formatted data from XML handler and send data to server

}

### goHome

{

go back to home activity

}

## SENDDATA

## NETWORKSERVICE

Handles data transfer from client to server

### onHandleIntent

{

get ip, host, and data information from sending activity

send data to server

}

## HELPERS

## GPSHELPER

Handles GPS location and listens for location changes

### getLocation

{

get and return last known gps location

}

### updateLocation

{

set up the gps location listener to get updates based

on time inverval and minimum location changes

}

### GPSlistener

{

when location changes (based on the parameters set from updateLocation)

broadcast notification for gps change

}

## XMLHandler

Handles XML formatting for server consumption

### XMLHandler

{

create framework for XML document

elements should be created based on the sample XML provided (See Sample XML)

}

### updateGPS

{

get information from gps location and update XML document

}

### updateTime

{

call getTimeFormatted and update XML document with formatted time

}

### updateIdent

{

get client details (ip, hostname, MAC) and update XML document (See Sample XML for needed info)

}

### updateInfo

{

get device details (imei, phone,.. etc) and update XML document (See Sample XML for needed info)

}

### clearNode

{

clears all child nodes in an XML element

}

### getStringFromDocument

{

update gps and update time and return formatted XML as a string

}