Developing A-GPS as a Student Project

Bahram Bahar, Adam Bolcsfoldi, Jonas Falkevik, Roger Jakobsen, Kristoffer Kobosko, Jimmy Källström Alexander Harju, Andreas Hasselberg, Johan Persson, Mattias Wadman
Olle Gällmo, Anders Hessel, Leonid Mokrushin, Paul Pettersson

Department of Information Technology, Uppsala University
P.O. Box 337, S-751 05 Uppsala, Sweden.
Email: {baba0967, adbo1405, jofa4567, roja5225, krko7365, jika8285, alha3869, anha0825, jope2553, mawa0881}@student.uu.se, {crwth,hessel,leom,paupet}@it.uu.se

Abstract. The project consists of ten fourth-year computer science students at Uppsala Universitet developing an A-GPS (Assisted-GPS) system. During the fall term of 2005 the students have developed a module for GPS-calculations in a GSM-network and an application that demonstrates a possible way of using the calculations module. This paper describes the design, the development process and the results of the project.

1 Introduction

GPS is a system designed for positioning objects on and around the earth. The system is based on one way communication where the satellite provides the GPS receiver with the satellite's position along with other information the can be used to decide the distance between the satellite and the GPS receiver. A benefit of this design is that there is now communication from the GPS receivers to the satellites, ensuring that the satellites can not be affected by the GPS receivers. The design however also has two great disadvantages: a receiever can not prompt a satellite for data as it is needed, but instead has to wait for the satellite to broadcast the necessary information. The other great disadvantage is that the information used to calculate the distance between the satellites and the receiver is much harder to retrieve than the lesser data of the satellite's position.

2 Preliminaries

2.1 GPS

The system is based on a minimum of 24 satellites that moves in 6 orbital tracks. Each satellite moves at approximately 2 km/s at an altitude of 20200 km, broadcasting signals continuously. It is known for every broadcasted signal, which satellite emitted it and at what time. Given the position of a satellite and a sufficient chunk of a signal, a GPS receiver can confine its position to

the surface of a sphere. This is calculated knowing how fast the signal travels coupled with the time at which the signal was sent by the satellite.

Other aspects that must be taken into consideration is the distortion associated with the signal traveling through the atmosphere and the distance the satellite has traveled since the signal was sent. The scope of possible positions is reduced for ever satellite this is calculadet. It requires suffcient information from four satellites to be collected in order pinpoint the position of a receiver.

2.2 Assisted-GPS

In an Assisted-GPS system, the terminal is aided during some of the stages of the positioning process. The aid provided depends on the terminal's needs. However, all A-GPS terminals receive information on which satellites that are above the horizon from an estimate of the terminal's position, calculated using knowledge of the GSM-network's infrastructure. This information can be hard for the terminal itself to receive from the satellites if the terminal doesn't have sufficiently clear sight to the satellites.

2.3 Erlang

Erlang [1] is a "concurrent functional programming language". It is designed to perform multiple distributed tasks concurrently. This is achieved by making processes lightweight by limiting them to solely using stack based memory. Erlang needs a run-time environment since it is by default an interpreted language. Open Telecom Platform [2] is a library developed by Ericsson AB that defines a large portion of Erlangs behaviour, since much of Erlang development is based on this library.

Mnesia Mnesia is a real-time distributed database management system, implemented in Erlang. It is especially appropriate for telecommunication applications and other Erlang applications with need of continuous operation and that exhibit soft real-time properties.

2.4 Project Course

The A-GPS project stems from a project course at Uppsala University. During the start of the course lectures about projects where held. The purpose of the project is to learn how to work in and plan a large project.

3 How A-GPS Works

3.1 The server

The A-GPS system adds a server to the existing GPS system. The server uses a GSM-network for communication with the terminal, and the Internet for retrieval of satellite position information. Therefore the A-GPS server must be placed in between a GSM-network and the Internet.

3.2 Traffic

The terminal to be positioned will first receive a message via WAP-push which contains information on where to connect. The terminal then connects using GPRS and the protocol SUPL (Secure User Plane Location), used for transmitting data between the terminal and the server.

When connected, information about the terminal's location in the GSM-network is collected by the server. Provided with that information, the server gets current GPS-satellite data relevant to the terminal's location. The server then sends assistance data to the terminal corresponding to its requirements.

Upon receiving the assistance data, the terminal starts to listen for satellites for retrieving pseudo-ranges, which are used to calculate the actual position. These calculations can be done either by the phone itself or by the server. If it is calculated on the server, the terminal sends the pseudo-ranges to the server for processing. If not, the calculated position is sent.

3.3 Assistance data

Assistance data contains information about the satellites that are above the horizon from the terminals point of view. There are thre advantages of receiving assistance data using GPRS over the GSM-network.

- GPRS is package based, where as the GPS-system broadcasts the corresponding information, reducing the risk of losing data.
- The GSM network has better coverage indoors, also contributing to a reduced risk of losing data.
- Only relevant information is provided, reducing the amount of processing for the terminal.

The protocol used for assistance data is the Radio Resource Location Services Protocol [3]. RRLP is a general protocol that can be used for many different purposes of mobile location communication.

4 System Description

4.1 A-GPS server

Assistance data retrieval Only a few of the required parameter in the RRLP-protocol are actually mandatory. It is the task of the developer determine what information the terminal needs for pseudo distance measurement.

The developed application uses Mnesia to store data about the satellites. The data is collected from NASAs ftp archive (cddis.gsfc.nasa.gov), which holds hourly updates of the satellites position as well as other data. This is done periodically at an interval of 30 minutes, to ensure the most recent data available is used. The data is then processed and put in the Mnesia database. Also stored is a table containing a geographical grid where every cell/record has a list of satellites currently visible from within that cell, calculated using the data collected from the ftp archive. This process runs every ten seconds to ensure that the data is correct.

Request handling The part of the A-GPS server handling requests from the MLC has two major components: the traffic manager and the calculation manager. Both are separate long running processes that spawn rocesses, called handlers, to do the actual work. In a distributed Erlang environment, the managers could easily be replaced by other handlers on different Erlang nodes when handling high traffic loads.

Traffic manager The traffic manager waits for positioning requests from the MLC. To be able to handle multiple requests at the same time it will spawn a separate traffic handler process for each new request.

Traffic handler The traffic handler takes care of the RRLP communication with the terminal and performs encoding, decoding and segmentation of messages. On initiation, it will call the calculation manager to start a new Calculation handler dedicated to this position request and ask it to calculate assistance data. On response, it will send the data together with a position request to the terminal. Upon receiving the position response from the terminal, it will, depending on response type, ask the calculation handler for position calculations and wait for it to response with position, or if the terminal did the positioning it will directly send the position to the MLC.

Calculation manager The calculation manager waits for requests from traffic handlers, and as the traffic manager it will also spawn seperate calculation handlers to handle the requests.

Calculation handler The calculation handler takes care of all satellite and GPS related calculations. Upon being called, it waits for assistance data or position measurement requests from its traffic handler.

A-GPS server environment simulation The SMLC Simulation Tool main purpose is to measure the performance of the A-GPS server. This is done by, during some time interval, creating a large number of positioning requests and measure the response time for each. The time for each step in the positioning is measured, except for the data transfer to and from the terminal and the time it takes for the GPS-chip to locate the satellites. These values are instead simulated with fixed values that either can be configured before and/or changed in run-time.

The results of a simulation can be logged to a file and the current status of the benchmark can be printed to the console. This information can be printed continuously at a configurable interval that can be changed in run-time. Functionality for changing the interval of positioning requests to at least one per millisecond, and for changing the positioning method between MS-based and MS-assisted has also been implemented. The number of visible satellites for the terminal can also be changed in run-time.

For each positioning request, the number of satellites in view for the terminal, the positioning method, the time spent for each step of the positioning and the time spent in total is logged. The current number of position requests and the current number of positions received is continuously printed to the console. As well as the current minimum, maximum and average positioning time. Provided

that the number of requests does not grow at a higher rate than the number of responses, the system is stable. Because of this important characteristic, also the difference is printed to the console each interval.

The simulator is implemented as a finite state machine and simulates both an SMLC and a terminal. A new process is spawned for each position request. This approach is made possible thanks to lighweight property of Erlang-processes. The logging is performed using events and an event manager that can log to either the console, a file or both. The simulator acts as, and consequently uses the same protocols for communication, as the MLC. Further information is available in the the "Communication with MLC"-section.

Communication with MLC An application initiates a positioning request byt sending a WAP-push SMS to the MLC, prompting it to connect to a specified address and port. Once the connection is established the SUPL protocol is used for information exchange. The calculation of position on the A-GPS server is started by the MLC using a remote procedure call.

After a link has been established, all communication is done with an interface using process communication. Information is packed with the Radio Resource Location Protocol (RRLP). The MLC puts all RRLP messages sent from the A-GPS server in a SUPL message, and forwards it to the terminal. The simplicity of process communication in Erlang enables distribution of the A-GPS server and the MLC on different nodes without effort.

The RRLP protocol specifies five different messages including "Measure Position Request", "Measure Position Response", "Assistance Data", "Assistance Data Acknowledgment" and "Protocol Error". "Measure Position Request" is sent from the A-GPS server to the terminal and includes assistance data. It may be preceded by up to seven "Assistance Data" messages, due to the maximum size of a message. The terminal responds to all "Assistance Data" messages with an "Assistance Data Acknowledgment". The terminal responds to a request with a "Measure Position Response"-message that includes either a pseudo range information or a position, depending on the method of positioning. In either case, the terminal might respond with a "Protocol Error" indicating an error with the data sent.

Calculations The calculations that are implemented in the A-GPS server can be divided into three different groups.

- Matrix and Vector calculations library As most other 3D calculation systems, the calculations the system is dependent on matrix and vector algebra in order to calculate positions, distances and angles. The matrix algebra library contains several functions, ranging from simple functions such as one that creates a new matrix to more complex ones, e.g. one which calculates the inverse of a matrix through the method of gauss elimination. The vector algebra library contains functions of which many are used by the matrix algebra library.

– Geodetics and angles The geodetics and angles calculation and conversion modules are used in order to wander between the different coordinate system formats used in the system. Currently the system uses a number of different formats, namely, cartesian, radian-based ellipsoidal and degree-based ellipsoidal. These formats are based on the same system, the WGS84 ellipsoid. Further conversions and transformations to systems based on other ellipsoids are in the module transforms. These systems however are not used by the position calculation itself, but rather to convert the position so that it fits the maps used in Trace tool.
– Assistance data calculations In order to cut down the satellite signal search time for the Mobile Station, functionality for checking whether a satellite is in view or not has been implemented. It also checks the health of the satellite. The The satellite's positions are cached in the database and then used to determine which satellites are visible in each grid. The Assistance data calculations module uses the vector and matrix algebra modules to a very high extent.
– Position calculations The most complex function of the whole calculations part of the system is the one that calculates the position of the receiver. It uses almost all other functions in the modules. The algorithm behind it uses the least squares method on matrices in order to calculate a correction to an initial 'wrong' position in X, Y, Z dimensions and a time error. By iterating over the difference between each time the error is corrected, a fixed point is reached. And the final correction is added to the position. As inputs the function takes a set of satellite observations data. The algorithm heavily depends on the matrix, vector and geodetic modules.

4.2 Trace Tool

Trace Tool was designed to be used for demonstrating and testing the A-GPS server. It consists of a map renderer, a map database and a tracer.

Map-renderer The map-renderer's main task is to convert a GenMap-file (General Map format) to Scalable Vector Graphics (SVG) and Portable Network Graphics (PNG) files. Batik, a SVG-package for java, was used to implement the SVG generation and manipulation. The SVG can be used to generate PDF-documents. The PNG:s are used to generate images that are intended to be displayed on the web interface. The GenMap-files are adapted to the limited resources of the mobile appication.

Map database generation The map database stores map data in three formats, SVG, PNG and GenMap. The database stores complete tilings of images of the world at multiple zoom levels. The images are saved in files and in paths in a way that they can be identified by what area they cover.

Map database usage The requests to the map database consitsts of three fields: a centerpoint, a zoom-level and an image-format. The image files corresponding to the portion of the map requested is extracted, tiled properly, and returned.to build a map.

Map conversion Lanmäteriet and other parties delivering geographical data do so using one of the two common GIS formats ESRI Shape or MapInfo. This implementation of Tracetool focuses soley on ESRI Shape since it is the only format Lantmäteriet uses. ESRI Shape is a relatively primitive binary format. It uses shape files, containing the geographical data set, and database files (in dBase format) containing so-called Attribute data, i.e. miscellaneous information about each geographic entity in the shape file.

Furthermore, ESRI Shape uses a complicated system of varying bit order in all of its fields. Since no Java API for handling ESRI Shape input data was availiable, it was decided that the C API Shapelib was to be used. In a Java project such as Trace Tool, this means the neccessary bindings between C and Java code need to be provided. The approach was to use Java Native Interface (JNI), which would link the C-code of the importer, compiled as a dynamic loadable library, with the Java Virtual Machine.

The importer, called libgenmap, uses a filter file in a readable XML format to translate the shape and attribute data to a generic map format in XML called GenMap. The filter file sets the rules for the conversion. Since Shape data can exist in every language rules for translating element names into english as well as drawing data, such as stroke width and color in the filter file, is provided. By making it possible to create one filter file per shape file provider, it is ensured that Trace Tool works with any provider of geographical data, always generating map data of the same kind for use of Trace Tool, as long as the ESRI Shape standard is complied with. After the translation process, the resulting map data is delieverd to Trace Tool in the generic XML-based GenMap format, where it is added to the geographical database.

Tracing of subscribers Traces are iplemented by running separate threads for each tracing, that each make positioning requests at a fixed interval. A trace-thread runs during the whole duration of the trace and stores each successful positioning in a database. The traces can then be retrieved by the GUI and plotted as a path on the map or exported for external analysis.

Mobile application The mobile interface is an elementary positioning application developed for the J2ME [4] MIDP 2.0 [5] platform. It displays the position of an AGPS-client along with a map of the surroundings. The view can be navigated by zooming and scrolling. The position is retrieved from an AGPS-server and the map data from a specialized server.

Web interface The web interface contains functionality for managing subscribers as well as starting, stopping and showing traces. The interface commu-

nicates soley with Trace Tool using XML. It uses sockets to send commands to Trace Tool, and receives either the requested information or a status code of the requested command.

The web interface is a single user interface and therefore does not handle different interface users. The web interface is developed in Java and generates XHTML. A signle file contains all text to be displayed in the user interface, which provides an easy way of changing the language.

Command line interface A command line interface for Trace Tool has been implemented to control the core features of the system. In addition to the commands available in the web interface, the command line interface enables functionality for map generation and generating test pictures.

5 Evaluation and Testing

5.1 Evaluating calculations libraries

For evaluating the correctness of the calculations library, a test server was used. The test Server runs test suites, written in Erlang so that every function constitues a test, and logs the result to text and HTML files. During testing a few bugs where found and the error management where looked over.

5.2 A-GPS server

The A-GPS server's external interfaces were tested by using the simulation tools. The quality of computation, computation performance and RRLP error detection was tested.

6 Conclusion

The A-GPS server The implementation of the A-GPS server can on a whole be regarded as a success. Even though one part of the assistance data can't be determined properly the server's performance exceeded our expectations. The A-GPS server meets the performance requirements thanks to the supervisor/worker implementation. The communication between the MLC and terminal was implemented successfully.

Calculations All vector, matrix, and positioning algorithms were implemented and tested. When a complete positioning was made using real data in the correct format, the retrieved position was within the required margin of error.

The simulation tools The SMLC simulation tool was used both as a test tool of concurrent communication and performance of the A-GPS server. In the future, it will be possible to use the SMLC simulation tool to measure the response time of a complete positioning communication sequence of a live A-GPS server. Due to lack of terminals that supported SUPL communication the WAP-push component weren't properly tested.

Trace tool During the early stages of the project the purpose of the application part was more or less arbitrary. Consequently, it was decided that a map display and positioning was to be developed. It was to consist of a web application and a mobile application that communicate with a map data server. The mobile application was decided to be bare bone, while the web application should offer account features, allowing the user to store and track mobile phone number etc.. These ideas were however revised roughly halfway through on account of Mobile Arts requesting a tool for tracing positioning. The trace tool now implements much of the functionality requested by Mobile Arts.

References

1. Joe Armstrong, Bjarne Dacker, Thomas Lindgren, and H å kan Millroth. Open-source erlang - white paper. *http://www.erlang.org/white_paper.html*.
2. Seved Torstendahl. Open telecom platform. *http://www.ericsson.com/ericsson/corpinfo/publications/review/1997_01/files/1997012.pdf*, 1997.
3. The 3rd Generation Partnership Project. 3ggp ts 44.031 v7.2.0. *http://www.3gpp.org/ftp/Specs/html-info/44031.htm*.
4. Sun Microsystems. Java 2 platform, micro edition. *http://java.sun.com/j2me*.
5. Sun Microsystems. Java 2 platform, micro edition. *http://jcp.org/aboutJava/communityprocess/final/jsr118/*.