Hybrid Positioning Framework for Mobile Devices

Anja Bekkelien, Michel Deriaz
Institute of Services Science
University of Geneva
Switzerland
{anja.bekkelien, michel.deriaz}@unige.ch

***Abstract*—A variety of technologies has emerged in response to an increasing demand for location aware applications. Nevertheless, single-technology systems have several limitations and vulnerabilities and it seems unlikely that such a system will be able to provide a universal solution. In this paper, we present the Global Positioning Module (GPM), a framework that seamlessly combines a multitude of approaches in order to supply mobile devices with indoor and outdoor positioning. The novelty of our work is the way in which position providers are integrated by using an abstraction derived from their performance properties. This allows for a selection of providers based on their suitability to the surrounding environment and to the user's requirements with regards to accuracy, drift, power consumption and so on. Our aim is to provide a foundation for ubiquitous location based services, namely a transparent transition between the plethora of technologies available today.**

Keywords-hybrid positioning; model; mobile devices

I. INTRODUCTION

Numerous indoor positioning systems have been proposed. Most rely on a single technology such as WLAN, RFID or Bluetooth. This approach renders the system vulnerable to service failure and fails to take advantage of positioning information made available by other sources. Additionally, the coverage of systems depending on external infrastructure is limited to the areas in which this infrastructure is available, potentially necessitating the deployment of new, costly hardware. These issues have led to the development of hybrid systems that combine different technologies [1] [2] [3] in an attempt to increase robustness and cost-effectiveness. The use of RF technology together with pedestrian dead-reckoning [4] is an example of a type of hybrid system that has received attention. While these systems improve performance to a certain degree, they are still limited to a specific set of technologies. What is needed is a generic, technology-independent framework capable of exploiting the entire spectrum of available information.

Several high-level architectures have been proposed in the areas of context awareness and sensor data fusion (see [5] for a survey), in which positioning often has an essential role. One of the first and most influential models in this area is the JDL data fusion model created by the US Department of Defense's Joint Directors of Laboratories in 1985. It breaks the data fusion process down into 5 levels, each dealing with an increasingly higher level processing of output originating from sensors in order to, in the final level, do situation assessments based on the data. An example of a context-aware system using the JDL model for fusion of location information is presented in [6].

While such models are a useful instrument in the design of software systems, they are highly abstract and provide little guidance for the actual implementation. We address this issue by introducing a positioning framework based on an innovative model for the harmonization of positioning technologies, and provide specific implementation details concerning the harmonization process and the unification of diverse sensor and location service output data. Our solution already supports a wide range of commonly used technologies and provides an interface that allows developers to easily add others in a plug-in fashion.

II. RELATED WORK

In an attempt to standardize an ever increasing amount of positioning technologies, several frameworks and models have been proposed. Early work in this area includes the Location Stack [7], providing a multi-layered abstraction for location-aware ubiquitous systems, and its implementation [8], in the form of a flexible framework that successfully integrated three different positioning technologies.

A technology and application independent framework was proposed in [9]. They describe a five-layered context model used for processing low level sensor output into high level locations and present an implementation based on a distributed server-client architecture. They state that a generic location handling framework has four requirements, namely a layered context model, a distributed architecture, a technology independent and extensible location format and a technology independent programming interface.

Kritzler and Krügler [10] present four challenges related to tracking using data from heterogeneous sensor sources. They addressed the challenges by developing geTrack, a tracking system capable of using input from different sensors on a device. A data model for the storage of sensor output data is proposed. Two use cases in which their system is used are presented, for each of them they employed different data analysis methods.

iPOS [11] is another multi-technology positioning architecture, developed specifically for mobile devices with limited resources and off-the-shelf sensor hardware. It has an

open plugin architecture, allowing for easy integration of sensor plugins, making it robust and scalable.

Collaborative positioning is proposed in [12] as a way of calculating positions based on a network of multi-sensor devices in order to provide indoor and outdoor positioning.

A recent effort is the distributed indoor location system able to combine multiple technologies proposed by Martínes, Villanueva, Santofímia and López [13]. Their solution consists of two sub-systems, a Location Event Provider and a Location Event Consumer, separated using an object oriented middleware. A high level interface is defined to provide users of their service with a unique way of handling position information.

In [14], an approach to integrating multiple indoor/outdoor positioning technologies in a smartphone is presented. The solution uses multi-sensor, satellite and terrestrial techniques, with an architecture based on the Model-View-Controller pattern.

Our work differs from the previous work in that we provide specific definitions for how to abstract different technologies. Where the previous work is focused on the general framework architecture and the abstract layers of data processing, we are concerned with providing precise details for the implementation of such frameworks.

III. METHODOLOGY

Our solution was developed with the aim of providing users with a universal positioning framework capable of tracking locations in any setting, both indoor and outdoor, and which satisfies the following requirements:

- *A single, uniform interface towards all positioning technologies.* Technology-specific details should be hidden from the users. The system should provide a single point of access for obtaining position estimates.
- *A standard representation of geographical positions.* This is a consequence of the first requirement. Different technologies are likely to return information in different formats, and in order to provide a consistent interface, the information must be processed into a uniform representation.
- *Deployable on commercial off-the-shelf mobile devices.* The framework will primarily be developed for the Android platform. It must be able to function on devices with limited capacities, for instance those lacking certain sensors, and must manage its power consumption to avoid draining the batteries too quickly. In addition, the system should not be dependent on external hardware devices. Such devices may be used but must be optional.
- *Extensible.* It must provide users with an easy way of including other technologies so that it can be customized to fit their needs.
- *Adaptable.* The system should be aware of its environment so that it is able to adapt to the context in which it is used.

We first introduce the concept of a *position provider* as an abstraction of a component, either hardware or software, that is able to provide its geographical position. The abstraction describes the component solely in terms of its performance in different settings using a set of criteria, therefore eliminating the large differences between the various positioning technologies and allowing all position providers to be handled in a uniform way.

We then define a model that precisely specifies the properties of a position provider, and which serves as the basis of our framework. The model consists of two main components, a specification of the provider's criteria and a specification of the position estimates returned by the position providers. These are presented in more details in the following sections.

A. Criteria

The most important feature of the model is the criteria. They define the provider's expected performance in terms of accuracy, precision, drift and power consumption. In this paper, we use the definition of accuracy and precision given in [15]: accuracy is the average error distance, and precision is the maximum error distance for a given percentage of measurements.

The criteria's purpose is to allow different providers to be compared against each other by an algorithm, so that the most appropriate one can be selected based on the context and the user's requirements. Each provider has its own set of criteria, to which values must be assigned. Yet, it may not always be possible to obtain values for each criterion. In some cases, certain criteria may not even be relevant to the provider. Those criteria should be left unspecified. The values may be found empirically by conducting tests in the various settings in which the provider will be used. Data gathered during these tests are then used to compute the statistics about the provider's performance from which the criteria values are obtained. For providers based on commercial third-party systems, the values of certain criteria may be given by the manufacturer. Note that the criteria are static values.

Position providers will behave differently depending on the surrounding environment. It is therefore necessary to allow for the possibility to specify different profiles for each provider. An example where this would be useful is for specifying the expected accuracy of the GPS in an open landscape as well as inside a city with tall buildings, where the coverage would be poorer. We include this feature by defining a set of profiles for each provider. This allows for a highly adaptable implementation, capable of taking the environmental settings into account.

The criteria are defined as follows, and a summary is shown in table I.

- *Horizontal accuracy*. The mean error between the real and the estimated position, measured in meters.

TABLE I. CRITERIA FORMAT

Criteria	Units	Type	Default value
Horizontal accuracy	m	Float	NaN
Horizontal precision	m	Float	NaN
Horizontal distance drift	%	Float	NaN
Horizontal time drift	m/s	Float	NaN
Vertical accuracy	m	Float	NaN
Vertical precision	m	Float	NaN
Vertical distance drift	%	Float	NaN
Vertical time drift	m/s	Float	NaN
Priority		Integer	-1
Room detection	%	Float	NaN
Power consumption standby	mAh	Float	NaN
Power consumption request	mAh	Float	NaN

- *Horizontal precision*. Three discrete values from a cumulative probability function are used to represent the precision, namely 50, 80 and 95. A system that declares a precision of (10, 15, 30) has a location accuracy of 50% within 10 m, 80% within 15 m and 95% within 30 m.
- *Horizontal distance drift*. Distance drift of the system, measured in %. The drift is used for systems where the accuracy decreases with the distance, like for dead reckoning. For instance a drift of 20% means that the estimated position has an accuracy value augmented by 20 m after travelling for 100 m on a straight path. The drift value is measured like the accuracy (mean distance error).
- *Horizontal distance drift rate*. Time drift of the system, measured in meters per seconds. The drift is used for systems where the accuracy decreases with the time, even when there is no movement. An example of such a provider would be one based on inertial sensors.
- *Vertical accuracy*. Same as for horizontal accuracy but for the altitude.
- *Vertical precision*. Same as for horizontal precision but for the altitude.
- *Vertical distance drift*. Same as for horizontal distance drift but for the altitude.
- *Vertical distance drift rate*. Same as horizontal time drift but for the altitude.
- *Priority*. The provider's priority, 1 being the highest. If several position providers are able to return a position according to the user's request, then the one with highest priority will be chosen. For instance, if the user only asks for a position, while not giving any preferences like the power consumption or the required precision, then the one with highest priority will be chosen. The value is also used if during a request two or more providers cannot be selected by order of preference.
- *Room detection*. The probability of detecting the correct room, on the correct floor, in percentage. A room is defined as an area within a building enclosed by walls, a floor and a ceiling and having at least one entry point. The doors and windows may be open or closed. For providers based on electromagnetic radiation, their capability of room detection is influenced by the degree of signal attenuation caused by the construction materials of the building. The materials can be highly diverse, and include concrete, wood, glass and metal, all interacting differently with the signals. Providers capable of returning an exact location, such as barcodes, will have 100% correct room detection, provided that information about the geographical layout of the building is available. Techniques relying on fingerprinting should give the percentage measured when doors are open.
- *Power consumption standby*. The average power consumption in mAh of the provider when it is in stand-by mode, meaning ready to take a measure.
- *Power consumption request*. The average power consumption in mAh of the provider for each position request. The total power consumption of a provider is the result of power consumption standby + power consumption request * U / 3600, where U is the update rate in Herz (U = 0.1 means that a measure is done every 10 seconds).

Our choice of criteria is partially based on the performance characteristics of positioning systems defined by Gu et al. [15]. They state that the most important aspects of any positioning system are the accuracy and the precision. As our model is conceived for three dimensions, it is necessary to define them both horizontally and vertically. Drift, both in time and distance, is a vital criteria for any inertial sensor based system and is therefore essential to include in the model. We consider room detection equally important since in many cases, e.g. in private homes, a precise position is unnecessary as long as the system is capable of predicting the correct room. Room detection is fundamentally different from the accuracy criteria, since the size of a room is arbitrary. Finally, the power consumption criteria have been included since we are targeting mobile devices. Being able to judge providers on the basis of their power consumption is highly important for user satisfaction as applications that make heavy use of sensors are likely to quickly drain the battery if not monitored properly.

We have not included a criterion describing the delay involved in obtaining a position. The reason for this is that in most cases it is not a static criterion, but one that can be configured programmatically. Nor do we require the number of devices which can be located simultaneously to be specified. This would be a useful criterion in a distributed server architecture, but the model is intended for stand-alone applications which are themselves responsible for calculating positions. It is therefore not relevant.

B. Representation of Geographical Positions

We specify a general format for the provider output which includes the position, acceleration, orientation and speed. Because different types of technologies do not give the same kind of information, it is necessary to standardize the output before it can be used in a larger system. While some technologies, such as the GPS, provide absolute, three dimensional positions, others may provide coordinates relative to their own frame of reference. All providers are therefore required to transform the raw data received from the underlying technology into this format. It is not required that the provider gives all the values, but must provide at least one.

The format defined for the output is to the largest degree possible based on current SI standards. The justifications for the choice of units and dimensions for the various properties are outlined here. A summary of the format is given in table II, which includes the type, range and default values of the properties. The default value is used for uninitialized properties, i.e. the properties for which a provider do not return a value.

We define a coordinate system for the devices in order to be able to express rotation, acceleration etc. in three dimensions. The coordinate system is relative to the device, as shown in figure 1. This means that the axis follows the movement of the device. To give an example, the positive z-axis is orthogonal to the front side of the device, whether the device is lying face up or face down. The format has the following properties:

1) Position

The position is given in three dimensions using latitude, longitude and altitude. Latitude and longitude are defined using 7 decimal degrees, which gives each of them an accuracy of 1.11 cm at equator. As one moves away from equator, the distance represented by a longitude degree decreases due to the curvature of the Earth, leading to the longitude becoming more accurate the closer one is to the poles. The accuracy of the latitude remains the same independent of the distance to equator.

The altitude is defined as meters above sea level. Since there are areas that lie below sea level, the altitude may be a negative number.

The reason for using 7 decimal degrees is that we need to account for current and potential future positioning systems capable of offering centimeter-level precision. Examples of systems already capable of providing less than 10 cm accuracy are presented in [1] and include Active Bat, and the Beep system. A better accuracy (millimeter-level) has been deemed unnecessary for a system intended for human positioning.

2) Acceleration

The acceleration is defined as the rate of change in velocity over time, relative to free fall. In other words, it measures the force of translation in a given direction. It is measured in m/s^2, and the model supports acceleration in three dimensions in order to accommodate output from multi-axis accelerometers. Acceleration may be positive or negative, the latter indicating either that the speed is decreasing in the direction of movement or that speed is increasing in the opposite direction.

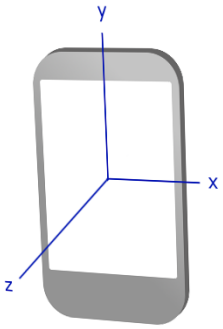

Figure 1. The coordinate system is specified relative to the mobile device.

3) Orientation

This parameter describes a device's orientation with respect to the Earth's frame of reference, and is represented by the devices' rotation around three axes. The orientation must not be confused with the direction of movement, because the two are not necessarily the same. An example of this is when the mobile device is held with the screen tilted 90 degrees while the user is walking.

The orientation is measured in degrees, using the standard definitions of pitch, roll and azimuth.

- *Pitch*. Describes forward and backward tilting, or rotation around the x-axis. The pitch takes on a value in the interval]-180,180]. When the device is lying such that its z-axis points upwards, the pitch is zero. The pitch is positive when the device is turned such that the z-axis moves in a clock-wise direction until it points downwards, and the device has been turned over, going from 0 to 180. If the device is instead turned such that the z-axis moves in a counter-clock-wise direction until the device is completely turned over, the pitch goes from 0 to -180.
- *Roll.* The sideways tilting or rotation around the y-axis. The roll will have a value in the interval]-90,90] and is positive when the positive z-axis rotates in the direction towards the positive x-axis, and negative otherwise. This can be illustrated with the following example: A device lying such that its z-axis points upwards, has a roll of zero. If the device is flipped 90 degrees to the right, in the direction of the positive x-axis, the roll increases to 90. If the device is flipped further, the roll decreases until it again reaches 0 when the device has been turned 180 degrees. If the device is instead flipped in the other direction, the roll values will be negative.
- *Azimuth.* Represents sideways turning, and is the equivalent of the compass direction. In other words, it is the rotation around the z-axis. Azimuth is measured in degrees east of the geographical North Pole and its range is defined as [0,360[.

4) Speed

The speed is measured in m/s.

TABLE II. GEOGRAPHICAL POSITION FORMAT

Property	Dim	Units	Type	Range	Default value
Position	Latitude	degrees	Float	]-90,90]	NaN
	Longitude	degrees	Float	]-180,180]	NaN
	Altitude	m	Float	]-∞,+∞[	NaN
Acceleration	X	m/s^2	Float	]-∞,+∞[	NaN
	Y	m/s^2	Float	]-∞,+∞[	NaN
	Z	m/s^2	Float	]-∞,+∞[	NaN
Orientation	Azimuth	degrees	Float	[0, 360[	NaN
	Pitch	degrees	Float	]-180,180]	NaN
	Roll	degrees	Float	]-90,90]	NaN
Speed		m/s	Float	[0,+∞[	NaN
Accuracy	Horizontal	m	Float	[0,+∞[	NaN
	Vertical	m	Float	[0,+∞[	NaN
Timestamp		ms	Long	[0,+∞[	-1
Provider			Integer	[1,+∞[	-1

5) Accuracy

The accuracy of a position, measured in meters. We represent accuracy horizontally and vertically instead of using the x, y and z axis. The reason for this is that while there are systems, such as GPS, that have different accuracies in altitude versus latitude and longitude, we know of no systems that have defined differences in accuracy for the two horizontal dimensions, nor do we see it as likely that future systems will inhibit such properties.

6) Timestamp

The timestamp marks the point in time when the position was recorded. It is defined as the number milliseconds elapsed since January 1, 1970, 00:00:00 UTC.

7) Provider

The provider property is the identifier of the provider from which the output originated. This is an arbitrary value, and serves as a reference to the provider's criteria.

IV. ARCHITECTURE AND IMPLEMENTATION

In the following, we present the Global Positioning Module (GPM), a framework implemented on the basis of the model. It was developed specifically for Android mobile devices, a platform chosen because it is free and open, and allows applications to run on a variety of devices, including smartphones and tablets.

GPM is an independent stand-alone framework that runs on a single mobile device. It does not depend on external servers or hardware but allows such entities to be used as an optional source of positioning information. This ensures true ubiquitous positioning. The framework is highly configurable thanks to the criteria, and allows the user to specify its behavior in different contexts. More specifically, it allows users to define the conditions under which a particular provider is used. As an example of how this is useful, consider an application which automatically changes to a provider with low battery consumption when the device's battery level reaches a certain limit. This is an important feature that lets the users adapt the system to fit their requirements.

In the design we have aimed for a system that is robust, adaptable and extensible. It consists of two main parts, namely a kernel and the collection of position providers. Figure 2 shows an overview of the architecture and the data transfer from the information sources to the providers and to the kernel. The figure shows a simple case where each provider uses a single data source but the frameworks also allows for providers to combine multiple sources. The framework supports several technologies, and the users can extend it by writing their own

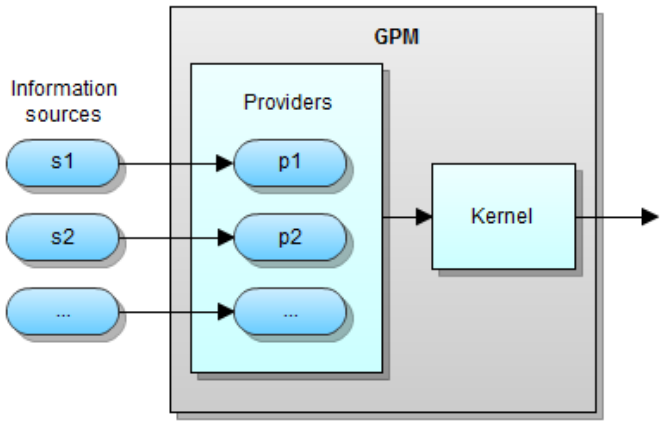

Figure 2. Transfer of position information from sensors and hardware components (information sources) to the providers and the kernel.

position providers. As long as it adheres to the specified interface and output format, any technology can be included.

A. Kernel

The central part of the system is the kernel which contains the specification of the position provider interface and the definition of the output format. This is also where the criteria are defined, as well as the classes that make up the main structure of the framework. Its design comprises an event listener system that is used by the providers to transfer processed position data to the framework's main class. This class is the entry point to the framework and is responsible for the creation and configuration of individual providers. The result is a simple, yet powerful structure for combining output from several loosely coupled sources.

The position provider interface specifies the operations allowed on the providers, which consist of methods for obtaining positions and for manipulating the providers' configuration. It guarantees that all providers can be accessed in the same manner and that their output has the same format. This is the key feature that allows the kernel to handle the providers uniformly. The implementations of the interface are responsible for transforming the raw data coming from the underlying technology into valid position estimates and for transferring them to the listeners, a process illustrated in figure 3. The actual transformation is technology dependent, and must be written specifically for each implementation.

A provider can be a wrapper for a hardware component on the device but it can also be entirely in software, as a wrapper around other providers. Using this approach, the lower level providers can be combined in accordance with the user's requirements and needs. It results in a hierarchical tree structure where the leaf nodes represent the physical components of the device, and the internal nodes define the relationship between them.

The kernel also handles the selection of the current provider. This takes place in a dedicated component that monitors the state of the mobile device and evaluates it according to a set of logical conditions before choosing a provider. As only one provider is used at any time, the component does not need to handle time synchronization of output coming from different providers. Where it is necessary to combine output from several sources, it is implemented using combined providers. Synchronization is therefore handled individually on a provider level.

A simple illustration of how this functionality could be used is the case where the most accurate of the available providers is always selected. Imagine an indoor environment where the main positioning technology is Wi-Fi, providing an accuracy of a few meters and covering the entire building. In certain rooms, however, a better accuracy is required, and has been implemented by adding Bluetooth beacons in these particular areas. The Bluetooth provider's criteria specify that its accuracy is higher than that of the Wi-Fi provider, so when the mobile device detects the presence of the Bluetooth beacons, it automatically switches to the Bluetooth provider. This is all transparent to the user of the mobile device on which the system is running.

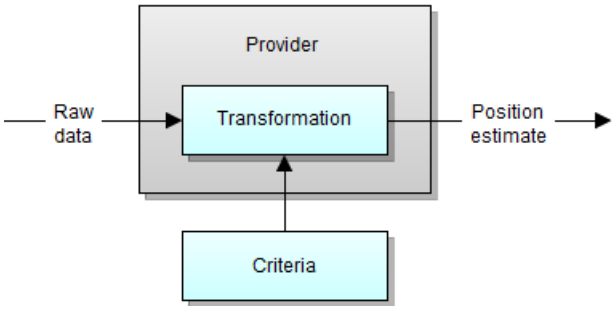

Figure 3. Transformation of position information.

B. Position Providers

The position provider component contains the various implementations of the position provider interface, of which we provide a brief description. We implemented a series of providers, ranging from sensor and hardware based ones such as dead reckoning, barcodes, GPS, Bluetooth etc., as well as various software providers, such as WsAltitude, which uses a selection of web services in order to estimate the user's altitude. Users wishing to extend the framework with their own providers should add them to this component.

We organized the providers by keeping each in a separate package. They all have a principal class containing the functionality specified by the interface in addition to any provider specific configuration properties such as update rates and server addresses. These values must be provided at the initialization of the provider. The following providers have been implemented:

- *Barcode*. Returns positions obtained using a barcode reader.
- *Bluetooth*. Uses fingerprinting by comparing a measurement of the signal strength of nearby Bluetooth beacons to a radio map.
- *Cell*. Provides the location of the cell tower to which the device is attached. Requires an Internet connection.
- *Dead reckoning*. Provides a horizontal position using step counting based on the mobile device's sensors.
- *EkoTek*. Based on the commercially available location tracking system EkoTek. Requires access to a dedicated server.
- *GPS*. A wrapper for the GPS of the mobile device.
- *GPSWsAlt*. Combines GPS with the WsAltitude provider in order to return a more accurate altitude.
- *IGPS*. An improved version of the GPS which uses trilateration based on Wi-Fi access points when GPS is not available. Requires an Internet connection.
- *Network*. A provider based on Androids' native Network location provider. The Network provider relies on cell towers and Wi-Fi access points to obtain positions. Requires an Internet connection.

- *Passive*. A provider based on Androids' native Passive location provider. It passively receives position information when other services request them. The provider itself never requests positions.
- *Wi-Fi*. Retrieves positions from a database of Wi-Fi fingerprints.
- *WsAltitude*. Returns the mean of the altitudes returned by various web services. Requires an Internet connection.

We created a small demo application, intended to aid developers who wish to include the framework in their own applications. It provides a simple graphical user interface for using and manipulating each of the available providers.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a novel approach for standardizing positioning technologies and position information. We first defined a model consisting of a set of criteria that serves to abstract different types of technologies. Then, we introduced GPM, a framework that makes use of this model in order to provide developers with a simple interface for adding positioning functionality to applications. This framework could serve as a low-level source of position estimates in context-aware applications or location based services.

The use of GPM provides several benefits. Most importantly, it gives developers a uniform way of employing a wide range of different positioning technologies in their applications. The ease with which new technologies can be plugged in, further emphasizes this advantage. To augment an application with position information, all that is needed is to include the framework, and initialize and start the position providers using simple method calls. Positions can be obtained either on the application's request or automatically as they become available, using the position listener interface defined in the kernel. In either case, the developer is shielded from all technical details.

Furthermore, the framework provides a standard way of defining geographical positions, saving the developer the trouble of dealing with highly dissimilar output data. The fact that it is developed for commercially available mobile devices ensures that applications based on it can be used by a large number of people, a group which will continue to increase as smartphones and tablets become more and more widespread. As the quality of the hardware sensors on these devices continues to improve, so will the accuracy of the framework's position estimates. Additionally, this ensures the possibility of providing seamless indoor and outdoor positioning.

Future work includes evaluation of the system with the aim of improving both the model and the framework. . In addition, we intend to extend the framework beyond the Android platform in order to reach a larger segment of users.

ACKNOWLEDGMENT

This work is supported by the AAL Virgilius Project (aal-2011-4-046).

REFERENCES

[1] S. Aparicio, J. Perez, A. M. Bernardos, and J. R. Casar, "A fusion method based on bluetooth and WLAN technologies for indoor location," Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on, pp.487-491, 20-22 Aug. 2008

[2] A. Baniukevic, D. Sabonis, C. S. Jensen, and H. Lu, "Improving Wi-Fi Based Indoor Positioning Using Bluetooth Add-Ons," Mobile Data Management (MDM), 2011 12th IEEE International Conference on, vol.1, pp.246-255, 6-9 June 2011

[3] Y. Gwon, R. Jain, and T. Kawahara, "Robust indoor location estimation of stationary and mobile users," INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies , vol.2, pp. 1032- 1043, 7-11 March 2004

[4] P. Kemppi, T. Rautiainen, V. Ranki, F. Belloni, and J. Pajunen, "Hybrid positioning system combining angle-based localization, pedestrian dead reckoning and map filtering," Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on, pp.1-7, 15-17 Sept. 2010

[5] W. Elmenreich, "A Review on System Architectures for Sensor Fusion Applications," SEUS 2007, LNCS 4761, pp. 547-559, 2007.

[6] A. M. Bernardos, P. Tarrio, and J. R. Casar, "A data fusion framework for context-aware mobile services," Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on, pp.606-613, 20-22 Aug. 2008

[7] J. Hightower, B. Brumitt, and G. Borriello, "The location stack: a layered model for location in ubiquitous computing," Mobile Computing Systems and Applications, 2002. Proceedings Fourth IEEE Workshop on, pp. 22- 28, 2002

[8] D. Graumann, W. Lara, J. Hightower, and G. Borriello, "Real-world implementation of the location stack: the universal location framework, " Mobile Computing Systems and Applications, 2003. Proceedings. Fifth IEEE Workshop on, pp. 122- 128, 9-10 Oct. 2003

[9] R. van Eijk, A. Peddemors, A. Salden, J. de Heer, P. Määttä, and V. Haataja, "Handling Heterogeneity in Location Information Services," In Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference, Jan. 2004

[10] M. Kritzler, and A. Kru□ger, "Tracking framework for heterogeneous sensor sources," Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on, pp.1-10, 15-17 Sept. 2010

[11] J. Bohn, "iPOS: A Fault-Tolerant and Adaptive Multi-Sensor Positioning Architecture with QoS Guarantees," Sensor Review, Vol. 27, No. 3, Emerald Group Publishing Limited, pp. 239-249, 2007

[12] A. Pourabdollah, X. Meng, M. Jackson, "Towards low-cost collaborative mobile positioning," Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 2010, pp.1-5, 14-15 Oct. 2010

[13] M. Martínez, F. Villanueva, M. Santofimia, and J. López, "A Multimodal Distributed Architecture for Indoor Localization," In Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN'11), pp.1–4 (online publication, no pagination), 2011

[14] L. Pei, R. Chen, Y. Chen, H. Leppakoski, A. Perttula, "Indoor/Outdoor Seamless Positioning Technologies Integrated on Smart Phone, " *First International Conference on Advances in Satellite and Space Communications (SPACOMM), 2009*, pp.141-145, 20-25 July 2009

[15] Y. Gu, A. Lo, and I. Niemegeers, "A survey of indoor positioning systems for wireless personal networks," Communications Surveys & Tutorials, IEEE , vol.11, no.1, pp.13-32, 2009