Implementing Adaptive Mobile Notification Services: A Model Based Approach

William Brander, Janet Wesson
Department of Computer Science and Information Systems
Nelson Mandela Metropolitan University, PO Box 77000, Port Elizabeth, 6031
Tel: (041) 504 2323 Fax: (041) 504 2831
Email: {William.Brander, Janet.Wesson}@nmmu.ac.za

Abstract — Mobile notification services typically do not adapt to a user's physical context in terms of the mode of communication used for output. Implementations of these services are also not generally extensible and support only the service for which they were designed. This paper discusses the development and implementation of a model for adaptive mobile notification services which includes multimodality and context awareness and is extensible.

Index Terms— Context awareness, Mobile notification services, Adaptive models

I. INTRODUCTION

Mobile notification services can benefit from adapting to a user's environment [4]. Adaptation implies that the notification takes place in the most appropriate mode based on the user's environment. This adaptation can be achieved through the use of context awareness and multimodality.

Models that have been designed for adaptive mobile notification services are typically designed only for that particular notification service [5]. This paper proposes an extensible model that supports multiple notification services and that adapts to the user's environment.

Section 2 of this paper discusses some existing mobile notification models; Section 3 discusses the proposed model; Section 4 discusses the implementation of a prototype based on the proposed model while Section 5 focuses on future work.

II. RELATED WORK

There are a limited number of models for mobile notification services which support adaptation [5]. Most of these models, however, do not support extensibility. Previous research has identified a set of criteria that an adaptive mobile notification service should satisfy [3]. These criteria are:

1. **Abstraction** (context acquisition must be abstracted from context use).
2. **Flexibility** (new sensor technology must be supported without re-designing the model).
3. **History** (contextual history must be supported).
4. **Distributed Communication** (distributed communication between sensors, data stores and modules must be provided for).
5. **Synchronization** (multiple sensors must be able to synchronize their times in order to provide consistency).
6. **Mode switching** (the model must allow for the dynamic switching of modes).
7. **Mode independence** (the model must not rely on any particular mode in order to operate).

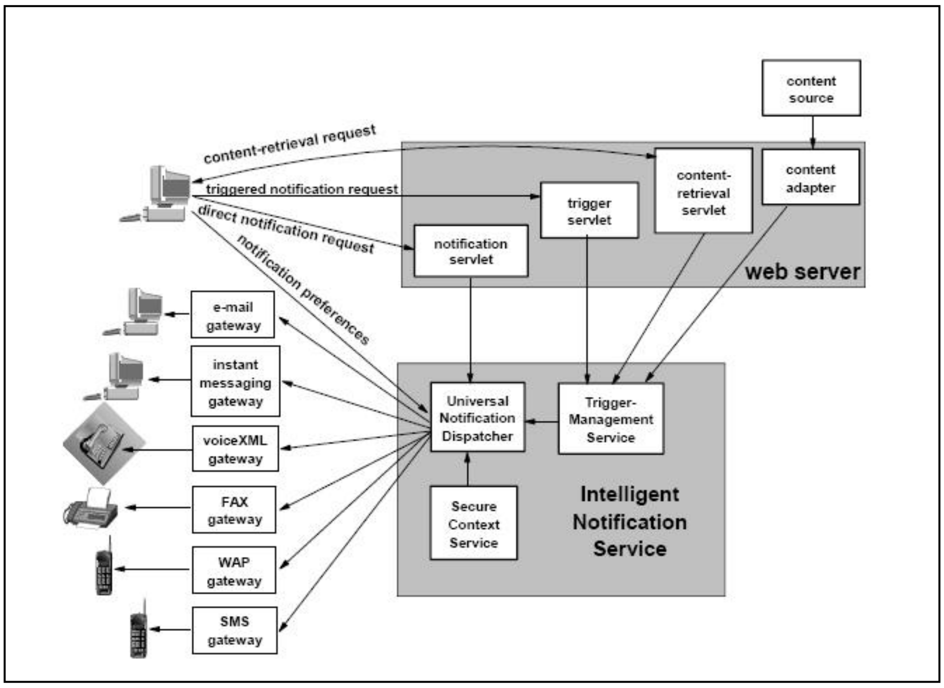
In order for a model to be extensible, it also has to support scalability, namely the ability to add new notification sources using a plug-in approach. This allows for new services to be added without changing the design of the model. The next sections discuss two models which support adaptation and extensibility.

A. IBM Intelligent Notification Service

The IBM Intelligent Notification System (IBM INS) [3] is an adaptive mobile notification service model which allows users to specify events which trigger notifications (Figure 1). These events may require the system to aggregate data from various content sources. The model is also designed to adapt to a user's environment in order to reduce the demand for user attention.

Notification can be made for various content types and sources, with each content source requiring its own content adapter which checks for new content and then formats any new content in a form suitable for use by the service.

The Content Adapter monitors the associated Content Source until new content is available. The Content Adapter then informs the Trigger Management Service of the new content. The Trigger Management Service then checks if any users are interested in that event. In this case, the Trigger Management Service requests a notification on the Universal Notification Dispatcher's notification queue. The Universal Notification Dispatcher then requests the user's



content source
content-retrieval request
triggered notification request
direct notification request
notification preferences
content-retrieval servlet
content adapter
trigger servlet
notification servlet
web server
e-mail gateway
instant messaging gateway
voiceXML gateway
FAX gateway
WAP gateway
SMS gateway
Universal Notification Dispatcher
Trigger-Management Service
Secure Context Service
Intelligent Notification Service

Figure 1. IBM Intelligent Notification Service [2]

context from the Secure Context Service and based on the users context and preferences, makes a notification to the user, in the user's preferred mode.

Requests for a user's context are passed to the Secure Context Service which uses multiple sensors to determine the user's environment. If a particular sensor is not available, the best approximation to the user's environment is returned. Each sensor is implemented in a gateway approach, allowing new sensors to be added and old ones removed without having to alter the design. As a result, the model is flexible with regard to the sensors supported; abstracts context acquisition from context use; and supports distributed communication between the various sensors.

The IBM INS does not, however, support mode switching as user input is limited to one mode per operation. Also, user registration is done solely in one mode (from a web interface) and the model is thus dependent on a particular mode.

From the above discussion, it can be deduced that the IBM INS satisfies all but two of the requirements for an extensible adaptive mobile notification service, namely mode switching and mode independence (Section II).

B. W3C Multimodal Interaction Framework

The W3C Multimodal Interaction Framework [11] identifies the major components that a multimodal system should possess and describes how these components relate to each other.

The input is provided by multiple input modes and devices. The input from each mode is captured and translated by a recognition component and the result is then interpreted. The output from each interpretation is combined by an Integration Component which determines that two inputs are related. This enables a user to select the option to delete something in one mode, and select the object to be deleted in another mode.

The combined input from all devices and modes is then passed to the Interaction Manager. The Interaction Manager takes the input and combines it with the applications functions, any session information and the system and user's environment information. The model does not cater for more than one service being implemented at once as multiple application components would be needed.

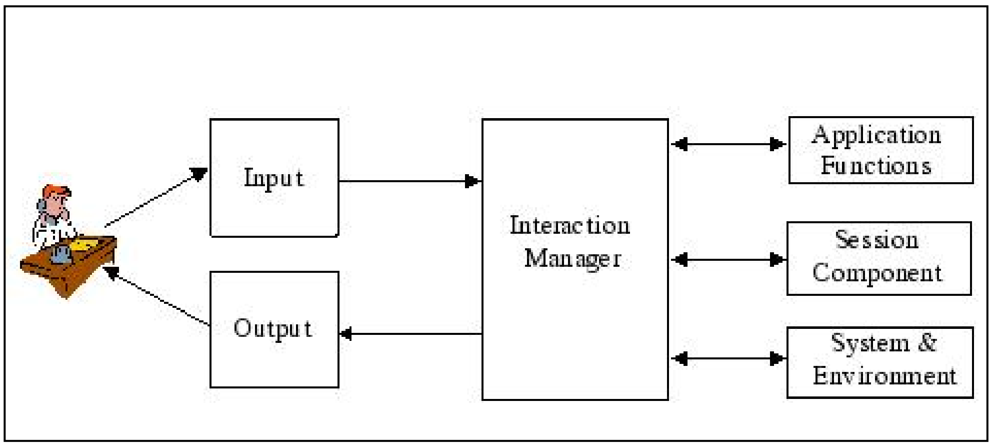


Figure 2. W3C Framework for Multimodal Interaction [7]

The System & Environment Component is designed to allow multiple sensors to be used in determining the user's context; but, specific sensors are not used in determining a user's context. This means that the model supports abstraction, flexibility, time synchronization and distribution of the sensors. The history of a user's context is not directly supported by the model, but could be included with a simple data store. The system and environment information is responsible for adapting the mode of output based on the user's context. After the Interaction Manager has finished computing an output based on the input, it passes the output to the Output Component.

The Output Component accepts an output string as input. This string is then processed by a Generation Component which determines which output mode will be used for the notification. The Generation Component may select a single, or multiple modes for output. The Generation Component passes the output to the various Styling Components associated with those output modes. Each Styling Component formats the output in a suitable presentation and then passes the formatted output to its associated Render Component. The Render Component is the component which actually presents the data to each user, be it in SMS format, or via a web page or using voice.

C. *Comparison of models*

Both of the above models support abstraction, flexibility, sensor distribution and time synchronization. They also require only minor changes to include contextual history. The differences between these models become apparent when you consider extensibility and multimodality. The IBM INS model fully supports extensibility, allowing for multiple simultaneous services while the W3C Framework supports only one service at a particular time. The framework is not designed for a particular service, and any content source could be used for notification, but only one service can run at a time unless the design of the framework is changed. The W3C Multimodal Framework allows for multiple modes and devices to be used as input for a command while the IBM INS model allows for only one input mode and device per command. A model which provides support for extensibility and complete multimodality has been designed by combining the IBM INS model and the W3C Framework. This model is discussed in the following section.

III. PROPOSED MODEL

The proposed model aims to support extensibility as well as to satisfy all the requirements for an adaptive mobile notification service. In order to be completely extensible, it draws from the IBM INS model and to support multimodality, it borrows from the W3C Framework for Multimodal Interaction. The result is a model which supports both multimodality and extensibility (Figure 3). The next sections discuss the different modules in the proposed model in more detail.

A. *Input Module*

The Input Module is responsible for taking input from the user, regardless of mode or device, and converting it into a standard format which is device and mode independent. The IBM INS model allows for only basic user input which has to be completed in the mode and on the device it is initiated. In contrast, the W3C Framework supports input from multiple modes and devices. The Input Module of the proposed model allows for user input of one command to span multiple devices and modes. Each device passes its input to a recognition component which translates the input into a form for further processing. Once the input has been processed, it is passed to an interpreter component which has the responsibility for converting the processed input into

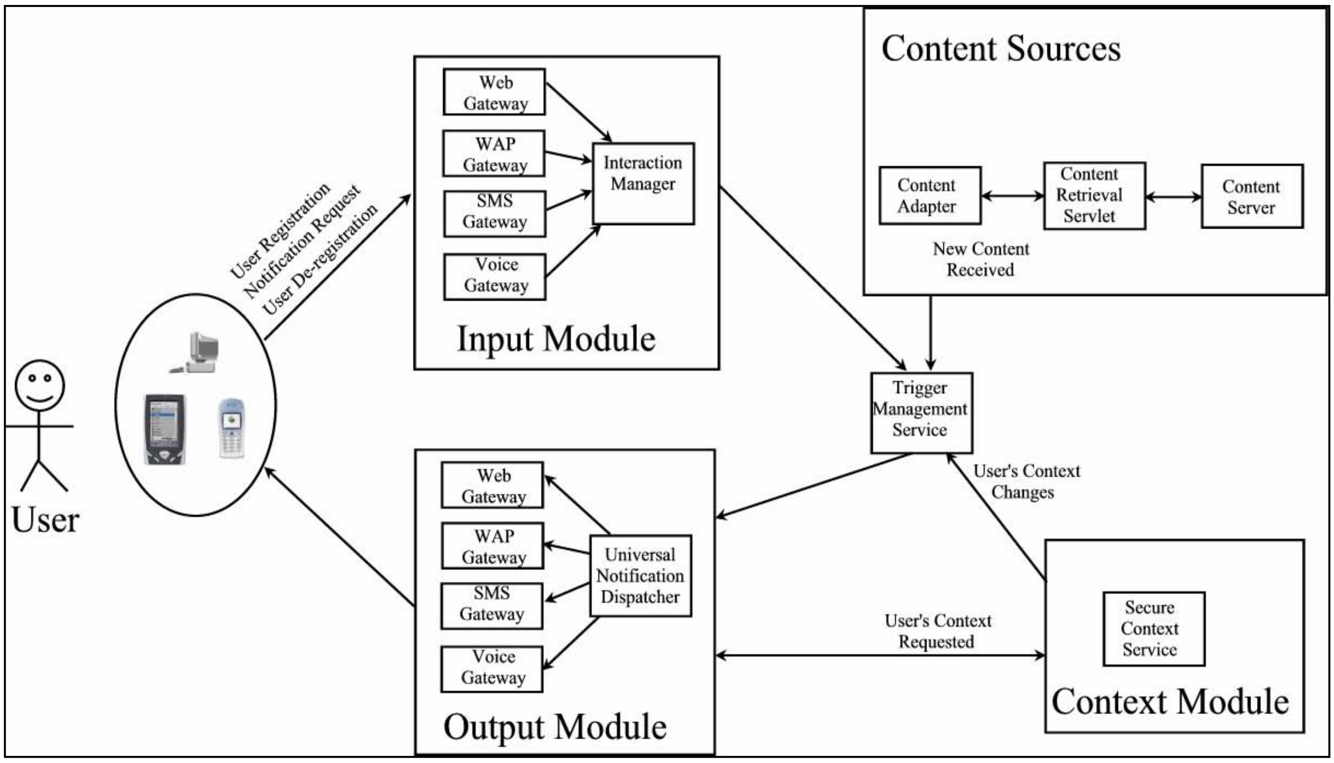


Figure 3. Proposed model for adaptive mobile notification services

a standard format used to represent modeless user input; typically a formal grammar will be used to specify this format. The converted input is then passed to the Trigger Management Service.

B. *Trigger Management Service*

The Trigger Management Service is the module that initiates the notifications based on events. The Trigger Management Service consists of a Trigger Manager, an Event Matching Engine and a set of Data Stores. The Trigger Manager maintains a list of Triggers that are running. Each Trigger is associated with a particular service event such as new email arriving on the mail server, or a web site being updated.

The Event Matching Engine receives an alert from a Content Adapter about the event that has occurred, as well as the information about the event. It then checks the Trigger Manager to see if there are any Triggers running for the event. If there are, the Matching Engine writes the event and its data to the Content Store and calls the specific Trigger's *handleMatch* method. Each Trigger's *handleMatch* method performs a task relating to that event; generally this will result in a notification being made, but it could also remove the Trigger from the Trigger Manager. This is to cater for those events that a user only wants to be notified of once, such as stock market prices. If a user wants to be notified if a stock price drops below a certain level, he/she might not want to be notified continuously if the price keeps dropping once it has reached that point.

C. *Content Sources*

The Content Sources are the various sources of content for each service, such as e-mail servers, news web sites or stock market sites. Each content source has a Content Retrieval Servlet which is responsible for monitoring the server for new information. Note that even though the notification is push-based, the Content Retrieval Servlet could receive its information from either push or pull methods. This means that a News Retrieval Servlet could constantly poll a news web site if there is no push notification available. Once new content is available, the Content Retrieval Servlet passes the content to the Content Adapter. Each content source has its own Content Adapter but each Content Adapter could be associated with multiple Content Retrieval Servlets. Once the Content Adapter has converted the content into the standard format for the Trigger Management Service, it passes the formatted content to the Trigger Management Service.

D. *Secure Context Service*

The Secure Context Service is responsible for determining each user's context and passing that information to the Output Module. The Secure Context Service is based on a combination of the Secure Context Service from the IBM INS and Dey e*t al's* context toolkit [9] and consists of various widgets, history stores and a Context Request Mediator. For the purposes of this paper, Context is defined as *any information the can be used to characterize the situation of an entity, where an entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves* [2] and a widget is defined as a small program which collects contextual information from a particular sensor. Each widget is responsible for collecting information from one particular sensor, but each sensor could have information collected from it by many different widgets. Examples of widgets are a location widget which is responsible for determining the user's location; a camera widget which uses a cell phone's camera to determine information about the lighting; and a microphone widget which detects the ambient noise level in the user's environment. The gateway approach of the widgets allows for new sensors to be supported by simply creating a new widget for each sensor.

In order to determine a user's context, as many sensors as possible are used. Combining the results from various sensors allows for a more complete interpretation of the user's context. For example, using the user's calendar information, a light sensor, audio clues to the user's environment and the user's location, it is possible to determine whether the user is in a meeting or out playing golf. Certain sensors are more applicable than others, for instance calendar information could be more relevant to a user's context than the air-pressure around the user. In order to cater for this, a weighting value is assigned to each sensor with a higher weighting implying more relevance.

Each history store stores time stamped information about each user's previous context information. This can then be used to determine extra details that would not otherwise be available, such as determining whether a user is walking or driving based on the rate of change of the users' location.

The Context Request Mediator is responsible for handling all requests for a user's context. It determines which widgets are supported by the user's device and uses a weighting function to combine the results from those widgets into a single, high level context. If a particular widget is not available, the best is done to determine the users context without this information. The Context Request Mediator also populates the history stores with each user's context, maintains all widget connections and manages time synchronization between the various widgets.

E. *Output Module*

The Output Module consists of a Notification Controller, a Preference Engine and a Notification Request Queue. The Preference Engine consists of a data store containing each user's preferences (For example user A is in context B, then notify using mode C), and multiple Service Preference Managers which hold notification preferences specific to each service. This allows a service like email notification to include preferences specific to the sender of the email. This allows notifications to be filtered based on content type and not just on the occurrence of events.

When the Notification Controller receives a notification request from the Trigger Management Service, it requests the user's context from the Secure Context Service and the user preferences from the Preference Engine.

Once the Output Module has the user's context the output mode selection is based on the user's preferences. If the user has specified that during meetings all notification must take place in a text mode and the context yields that the user is in a meeting, the notification will take place using text. The Preference Engine is responsible for selecting the user's preferred mode of output given the user's current context.

Once the output mode has been chosen, the notification along with the output mode is placed on the Notification Request Queue. The Notification Request Queue selects a message in the queue and based on the mode of output required and the devices available, selects the appropriate gateway to dispatch the notification. The selected gateway then sends the notification to the user. Since gateways have been used for output, it is easy to add support for new output modes and devices.

IV. IMPLEMENTATION

In order to implement a prototype based on the model in Section III, an incremental approach was followed. Implementing and testing each section individually encouraged a modular approach to the implementation. Email notification was chosen as the service to be used in order to validate the model. When a users' inbox receives a new message, the user will be notified of the message in the most appropriate mode based on the user's preferences.

Two key sections are the implementation of the Trigger Management Service and the Output Module. The Trigger Management Service is responsible for supporting most of the extensibility as additional services can be plugged into that component. The Output Module is responsible for handling the context awareness and generating the multimodal output. The implementation of these sections is discussed here.

A. Trigger Management Service

The Trigger Management Service is responsible for managing a pool of all triggers currently active and initiating all notifications. The Trigger Management Service is implemented in C#.

All notification services are implemented in a DLL which is then loaded by the Trigger Management Service. Within the DLL is a method that halts the current thread until a match has been made by the Event Matching Engine. Thus, each notification service has its own thread and is responsible for checking its own content sources. This means that content can be retrieved by either a push or a pull approach. The Trigger Management Service has a thread which periodically checks for a new notification service's DLL. This enables additional services to be added without restarting the service.

When a notification is necessary, the Trigger Management Service places the required notification message on the Notification Queue of the Output Module (Figure 3).

B. Output Module

The Output Module is responsible for sending all notifications. This involves finding the specific user's context; deciding which output mode to use based on this information; and selecting the appropriate output gateways to send the messages. The Output Module maintains a Notification Queue which contains a list of all messages that must still be sent to the users. The Output Module takes the first message in the list and sends a request to the Secure Context Service for that person's context.

The Secure Context Service uses multiple sensors to determine the users' context. Each sensor has a weighting assigned to it. To determine which context a user is in, a weighting function is used. For a particular context C, the weighting function processes all the sensors that the user's device supports. If a particular sensor supports the fact that the user is in context C, the weighting of that sensor is added to the total context sum. A total context sum is calculated for each possible context that the user can be in and the context with the highest total context sum, is the most likely context that the user is in.

Once the Output Module has the users' context it determines which output mode should be used based on the users' preferences (Section II.E). If the user has specified that during a meeting all notification must take place in a textual mode and the user's context indicates that he/she is in a meeting, text is used as the output mode. Each output gateway supports at least one output mode. Each gateway is also given a weighting. If more than one gateway supports the required output mode, the gateway with the highest rating is chosen. The output gateway is responsible for converting the message into whatever format is required. For voice output, this means employing a Text To Speech converter.

Output gateways are implemented as DLL's. This allows for additional gateways to be added without recompilation of the Output Module. The Output Module contains a thread, similar to the one in the Trigger Management Service, which periodically scans for new output gateways. This means that output gateways can be added at and during runtime.

V. FUTURE WORK

The prototype still needs to be deployed, but once it has been deployed the effectiveness of multimodal interaction in a mobile environment will be evaluated. Previous research has shown that the more a device knows about its environment, the more capable it is of supporting the user and the simpler the interface can be made [15]. A thorough user study can help confirm this theory.

User testing will be conducted to determine which sensors are useful in providing information related to a user's context. Considerable research has been done into location-aware systems [12], but it may turn out that location is irrelevant in determining a user's overall context. If it is known that lighting information, temperature information, location and the surrounding noise level is enough to accurately determine a user's context; device manufacturers could begin producing devices that support those sensors more efficiently.

The extensibility of the model will be proven during 2006 by an honours student at the Nelson Mandela Metropolitan University. This student will be implementing additional services, input/output gateways and context sensors.

Further research can also be done into using data mining to detect usage patterns of mobile notification services. If a notification always results in a user performing another task straight afterwards, the task could be built into the notification, thus simplifying the interaction.

VI. CONCLUSIONS

Mobile notification services have several benefits, but need to adapt to the user's physical context in order to operate effectively. Most existing models for mobile notification services do not, however, support adaptation or extensibility. The IBM INS model and the W3C Framework for Multimodal Interaction do support adaptation, but have several limitations (Section II).

This paper has proposed a model that meets the requirements for extensible mobile notification using context awareness and multimodality. The proposed model was based on a combination of the IBM INS model and the W3C framework (Section III).

The implementation of a prototype that demonstrates the concept of the proposed model was described. This implementation illustrates the extensibility of the model and allows for adaptation by means of context awareness and multimodality (Section IV). Further research will be done to determine the effectiveness of multimodal interaction in a mobile environment using user testing.

REFERENCES

[1] A. P. Afonso, M. J. Silva, "Dynamic Information Dissemination to Mobile Users," Mobile Networks and Applications, Volume 9, Issue 5, October 2004

[2] Abowd, G. C., Dey A. K., Brown, P. J., Davies, N., Smith, M. and Steggles, P., "Towards a better understanding of context and context-awareness", Handheld and Ubiquitous Computing, Springer, Berlin, pp. 304-307, 1999

[3] Bazinette, V., Cohen, N. H., Ebling, M. R., Hunt, G. D. H., Lei, H., Purakayastha, A., Stewart, G., Wong, L. and Yeh, D. L. An Intelligent Notification System. *IBM Research Report RC 22089*, 2001

[4] Brander, W., Wesson, J. L. Requirements for Adaptive Mobile Notification Services. Available Online: http://www.nmmu.ac.za/documents/WilliamBrander/RFAMNS.pdf, Last accessed January 2006

[5] Brown, P. J., Burleston, W., Lamming, M., Rahlff, O., Romano, G., Scholz, J. and Snowdon, D., "Context-awareness: Some Compelling Applications", International Symposium on Handheld and Ubiquitous Computing (HUC '00), 2000

[6] Chen, G. and Kotz, D. A Survey of Context-Aware Mobile Computing Research. *Technical Report TR2000-381*, 2000

[7] Cheriton, D., "Dissemination-oriented communication systems," Technical Report, Stanford University, 1992

[8] Costello, R. Mobile Technology in the Real Estate Industry. *Global Real Estate Now*, 2001

[9] Dey, A. K., Salber, D. and Abowd, G. D. A Conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human Computer Interaction*, 2001

[10] Dunnavant, S. Blackberries in Support of Technology. *Proceedings of the 29th annual ACM SIGUCCS conference on user services*, 2001

[11] Larson, J. A. and Raman, T. V. W3C Multimodal Interaction Framework. *Available Online: http://www.w3.org/TR/mmi-framework*, Last accessed February 2006

[12] Marcus, A. Mobile User-Interface Design for Work, Home, Play and On the Way. 2005

[13] Mitchell, K. A Survey of Context-Awareness. *Available Online: http://www.comp.lancs.ac.uk/~km/papers/ContextAwarenessSurvey.pdf*, Last accessed November 2005

[14] Munson, J. P. and Gupta, V. K. Location-based Notification as a General-Purpose Service. *WMC*, 2002

[15] Schmidt, A., Aidoo, K. A., Takaluoma, A., Tuomela, U., Laerhoven, K. V. and Velde, W. V. Advanced Interaction in Context. *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing (HUC '99)*, 1999

[16] Schmidt, A.,Beigl, M. and Gellerson, H. W. There is more to context than location. *Computers and Graphics*, 1999

[17] Steinfeld, C. The Development of Location Based Services in Mobile Commerce. *ELife after the dot.com bust*, 2000

William Brander is currently pursuing his MSc Degree at the Nelson Mandela Metropolitan University.