A Prototyping Environment for Investigating Context Aware Wearable Applications

Melekam Tsegaye[1], Shaun Bangay, Alfredo Terzoli

Department of Computer Science[2]
Rhodes University
Grahamstown, South Africa
g98t4414@campus.ru.ac.za
tel: +27 46 603 8641
fax: +27 46 636 1915

Abstract

In this paper we introduce the concept of a context-aware, wearable application prototyping environment, which can be used to support research into new wearable applications. We also present an initial specification for such an environment and show how different types of sensors can be modelled to produce data that describes a given context scenario using our prototyping approach.

Key words: context aware prototyping environment, sensor emulation, context emulation, wearable application prototyping

1. INTRODUCTION

A context aware wearable application prototyping environment can be used to support research into new wearable applications. In a prototyping environment, multiple software defined sensors that produce data streams according to a predefined behaviour can be assembled, the data they produce analysed, contextual information extracted from the analysed sensor data and the extracted contextual information fed to applications. In physical wearable systems, the task of assembling a variety of sensors can be difficult since time, effort and resources are required to build hardware implementations of sensors. Some sensors may not even exist at the time of investigation of a particular wearable application, thus such a prototyping environment not only acts as a wearable application prototyping environment but also as an environment where new types of sensors can be prototyped.

Wearable applications benefit from knowledge of the context in which they are being used in order to expand the range of useful services provided to users. The problem of providing context information to applications, so that they become context aware, has been tackled by a number of authors in the past [Schilit *et al*, 1994], [Pasco, 1998], [Brown, 1998], [Moran *et al*, 2001]. The context information we are referring to is that defined by Dey [Dey *et al*, 2001]

"*Any information that can be used to characterize the situation of entities (i.e. whether a person, place or object), that are considered relevant to interaction between user and application, including the user and applications themselves*"

Context information in wearable applications is an aggregation of external and internal sensory states such as environmental conditions, a user's psychophysical states, a user's location, objects/people a user is interacting with etc.

Implementations of a number of context aware applications exist [Korkea-aho, 2000]. Context information that is useful to a context aware application is acquired through a series of steps. Firstly multiple sensors are used to gather data about the state of entities that are important to an application. The data sampled by sensors is analysed and aggregated to form a picture of a past, present or future context. Collections of various context descriptors can then be used by context aware applications to make some higher level decision that depends on a given context.

Dey [Dey, 2000] provides a framework which context aware application developers can use, instead of having to re-implement operations that are common to all context-aware applications. Dey cites the following problems faced by context aware application developers

1) *Lack of a variety of sensors for determining a given context*: Most context aware applications tend to make use of a limited set of sensors, such as those that sense location [Schmidt *et al*, 2001].
2) *Difficulty of dealing with a variety of sensors*: Lack of support for programmers to work with specific hardware.
3) *Lack of a variety of context types:* Use of fewer sensors means availability of fewer contexts to applications.
4) *Inability to evolve applications*: Applications became difficult to extend and improve if they are

[1] http://www.cs.ru.ac.za/research/students/g98t4414
[2] http://www.cs.ru.ac.za

tightly coupled to specific hardware implementations.

Our approach, which uses a prototyping environment for investigating context aware wearable applications, addresses the above issues. The rest of this paper is organized as follows: in section 2 we describe previous works that are related to prototyping context aware applications, in section 3 we present a specification for a context aware wearable application prototyping environment, in section 4 we show how sensor data is modelled in our prototyping environment and in section 5 we present an example context scenario and show sensor data that is generated in our prototyping environment that describes the given example context scenario.

2. Related Work

Gu [Gu *et al*, 2005] proposes a middleware architecture for building context aware services. They advocate for a context model that uses a semantic web-based ontology [W3C, 2001] to share context information between various context aware services instead of using current representations of context information which use unstructured collections of text. They also advocate for the creation of context services that support the discovery, acquisition, interpretation and sharing of context information between various context aware services. Kwasar [Kawsar *et al*, 2004] proposes a hypothetical context aware application framework called Prottoy. Kwasar's framework is an attempt at generalizing the interactions of context aware applications with smart environments in which they will be used. Kwasar feels that context interpretation is application specific and does not integrate this facility in the proposed framework.

UBIWISE is a simulation environment for ubiquitous computing proposed by Barton [Barton *et al*, 2001]. UBIWISE is a proposal to provide a 2D and 3D environment where ubiquitous devices, such as digital cameras and cellular phones, can be modelled and user interactions with modelled devices can be studied. Barton has produced an example device modelling environment that uses the Quake III 3D game engine. Another project that attempts to provide a rapid prototyping environment for sensor based applications is the Liquid project at Lancaster University [Gilleade *et al*, 2003]. Liquid is a proposed universal interface for sensors which aims to allow the collection and representation of sensor data from a wide range of sensors and provide ubiquitous application developers a general purpose toolkit for developing sensor based systems.

The alternative to using a prototyping environment approach to investigating wearable applications is to implement sensors in hardware and create context aware services directly. A number of recent examples of these exist, such as Steve Mann's EyeTap system [Mann, 2004], Patel's ContextCam [Patel *et al*, 2004], Microsoft's SenseCam [Gemmell *et al*, 2004], and the PlaceLab smart home initiative [PlaceLab, 2004].

3. Specification for a Context Aware Wearable Application Prototyping Environment

Application developers already have modelling tools such as UML and integrated development environments (IDE's) that can be used to design and create new desktop applications. Context aware wearable applications are different from desktop applications in that they need to be aware of the state of their users and their environment. They provide services based on analysis of this information. Monitoring the states of users and their environment requires the use of a variety of sensors, and analysis of the data produced by sensors for determining users' context.

A context aware wearable application prototyping environment augments already existing application modelling tools and development environments by providing an environment where

- existing and new types of sensors can be simulated
- different types of contexts can be modelled
- wearable context aware applications can be prototyped

We propose that a context aware wearable application prototyping environment should at the least provide the following facilities: Sensor management, Context management, User modelling, Simulation management and Context Interpretation Management. We discuss each category in the following sections.

3.1. Sensor Management

The prototyping environment's sensor management facility should provide:

- *A method for defining a variety of sensors:* It should enable the description and representation of a variety of sensors.
- *A method for defining a variety of sensor data generation behaviours:* A sensor data generation behaviour definition is a specification of the type of data stream a sensor will produce when used in a simulation.
- *A repository for storing sensor and sensor data generation behaviour definitions.*

This facility is necessary in order for the prototyping environment to provide a variety of sensors that can be used as a source for sensor data which can be analysed to extract context information usable by context aware applications.

3.2. Context Management

The prototyping environment's context management facility should provide:

- *A method for defining a variety of contexts*: Contexts are defined in terms of the sensor data generation behaviour definitions of section 3.1.
- *A method for composing context scenarios*: Context scenarios are composed using previously defined context definitions.
- *A repository for storing context definitions and context scenario compositions.*

This facility adds a logical layer to the sensor data generation process. A collection of sensor data generation behaviour definitions can be used to describe a given user context such as *user is working in the office*, which refers to a user's *location context* and *mental state* context. To define this context, sensors such as those that can be used to determine the user's location, user's mental state and the devices the user is interacting with are necessary. A context scenario such *a day in the life of user* can be composed in terms of context definitions such as *user is working in the office, user is meeting with friends* and *user is shopping*.

3.3. User modelling

The user modelling facility of the prototyping environment should provide:

- *A method for modelling different types of user scenarios:* Users are modelled in terms of the context scenario that they are likely to experience using the context scenario definitions described in section 3.2.
- *A method for modelling different stories.* A way to model stories that users are likely to be part of. A story is composed using a collection of user scenario definitions.
- *A repository for storing different types of user models and stories.*

This facility of the prototyping environment is necessary to allow users to be modelled as participants in a variety of context scenarios that are of interest to context aware applications.

3.4. Simulation management

The simulation management facility of the prototyping environment should provide:

- *A method for the generation of sensor data*: Simulations of context scenarios are run using the user scenario models described in section 3.3 with the aim of generating sensor data that models a defined context scenario.
- *A method for visualising the data generated by a given sensor data generation behaviour*: This facility will allow the visual validation of sensor data produced by a given data generation behaviour definition.
- *A method for verifying that there is correlation between the different streams of generated sensor data*: When generating multiple sensor data streams there is a need to verify that there are no inconsistencies that would invalidate the context scenario they represent.

This facility of the prototyping environment is necessary to enable the running of simulations which produce multiple sensor data streams. The generated streams can be analysed by a context interpreter which extracts context information for use by context aware applications.

3.5. Context Interpretation Management

The context interpretation management facility of the prototyping environment should provide:

- *A method for interpreting context from raw sensor data streams:* Context interpretations are performed by analysing multiple streams of sensor data. The interpreted context information can then be aggregated and fed to context aware applications.

This facility of the prototyping environment is necessary to enable the extraction of context information from raw sensor data. It can also be used for experimenting with various context interpretation strategies.

4. MODELLING SENSOR DATA

Our implementation of the context aware wearable application prototyping environment specified in section 3 aims to provide all of the facilities described. This section discusses in detail how, in our prototyping environment, sensor data generation behaviours can be defined to produce sensor data that describe a given context scenario. Examples of real sensor data as well as modelled versions of the sensor data are also presented.

4.1. How sensor data generation behaviour is generated

The current implementation uses a selection of data generation functions with varying parameters. A piecewise function that is capable of generating a desired steam of sensor data is constructed through a visual editor. The function is then used to generate a stream of sensor data from a given range of data values. There are six data generation functions defined, all of which operate on any given data range. Their effects are shown in Table 1. At the moment these six functions are flexible enough to allow us to compose desired sensor streams.

Table 1: Sensor data generation functions

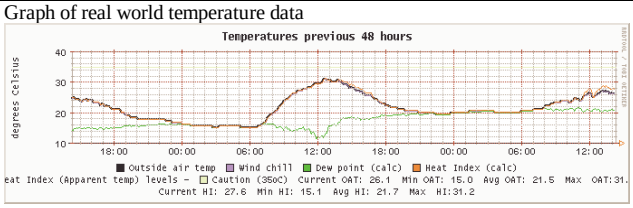
Function Parameters
data range – the range of data form which samples are produced
rate of change – value by which each value changes

Function	**Result**
Constant	Produces a constant sample value
Increasing	Produce samples values that increase in value
Decreasing	Produces sample values that decrease in value
X2	Produce sample values that are multiples of the function (1-(x*x)) -> [-1,inf)
Exponential	Produce sample values that are multiples of the function exp(x) x-> [-3,inf)
Random	Produces random sample values from the given range

4.2. *Two days worth of temperature data*

A temperature sensor is one of the simplest sensors to model. We obtained approximately two day's worth of temperature data from our local weather information service provider and modelled the same data in our prototyping environment. The result of our modelling is shown in Figure 1. The two graphs are very similar. The figure shows that we have produced a sensor data stream that resembles a real world one by using the pattern of real world temperature data as a base of our model.

Graph of real world temperature data



Graph of temperature data generated in our prototyping environment

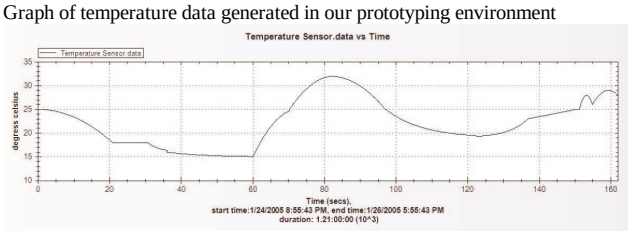
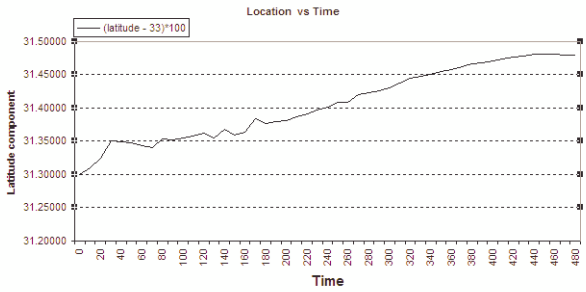


Figure 1 A comparison of graph of two day's worth of real temperature data and graph of temperature data generated in our prototyping environment.

4.3. *Location data of a typical user travelled route*

Location is one of the most exploited contexts in most context aware applications. We took a GPS unit and PDA on one of our regular routes on campus (8 minute walk for lunch to our dining hall) and recorded latitude and longitude data samples as we walked to our destination. We then attempted to model the same data using our prototyping environment. The results are shown in Figure 2. Only latitude coordinate values are shown for the sake of clarity.

Change of latitude component of location as recorded by GPS unit.



Latitude component of location data generated in our prototyping environment.

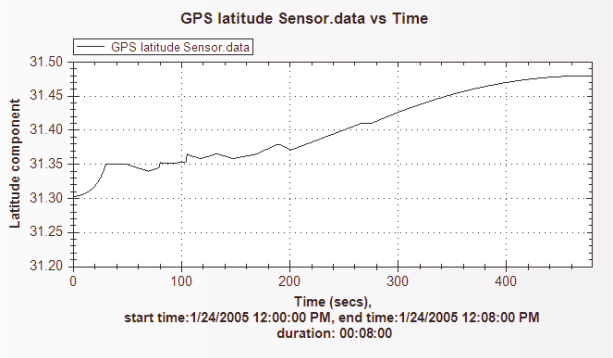


Figure 2: A plot of changes in latitude coordinates of our location on one of the routes we travelled on campus as well as the simulated version. We are located in and around 33.31 South latitude, the Figure shows a plot of the decimal component of our latitude coordinate, multiplied by 100, vs. time.

4.4. *Completely fictitious sensor data*

We mentioned in section 1 that one of the main advantages of using a prototyping environment to investigate context awareness in wearable applications is that the prototyping environment's sensor management facility allows us to define fictitious sensors that may not have real life equivalents. Figure 3 shows an example of a fictitious taste sensor defined in our prototyping environment which represents the taste of substances using a made up taste scale

Closer to 0 - Neutral
Closer to 1 - Sour
Closer to 4 - Fruity
Closer to 6 – Sweet

An implementation of a real taste sensor with a more complex scale does exist [Toko, 98]. Figure 3 shows the state of an individuals taste sensor (tongue?) over a period of 90 seconds.

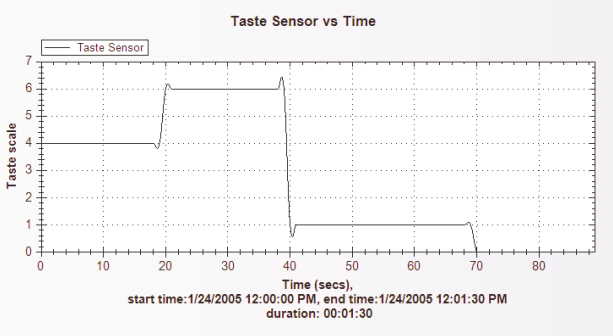


Figure 3: A fictitious taste sensor

5. Example Context Scenario

In this section we present an example context scenario that demonstrates how our prototyping environment can be used to model a given context scenario. A description of the scenario is shown in Figure 4.

Context scenario: Lunch time
Begin at 12.00 pm
Duration: approximately 1 hour

Context information definitions that make up the context scenario
Temperature was hot
Light level was bright (Sunny)
Location from residence to dining hall and back
Taste was neutral(Saliva) then changed while eating food then went back to neutral (Saliva)
Mood was varied calm, exhausted, agitated, nervous, excited, cheerful, laughing, calm
video (1 hour recording available)
People (number of people)
Internet signal available (excellent to good), (average to poor then none) not available, (excellent to good) available
User Movement (sitting, walking, sitting, walking, sitting)

Available Sensors
Temperature
Light level
Location
Taste
Mood
Video
People
Wireless Lan Sensor
Speedometer

Expected Result of Simulation
Data streams for each of the above sensors that describe related context definitions

Figure 4: Example context scenario: Lunch hour

Sensor data generation behaviours, that describe the related context for each sensor shown in Figure 4, were created using a combination of the data generation functions shown in Table 1 and stored as the *lunch hour* context scenario. The prototyping environment's simulation facility was then used to generate sensor data from the *lunch hour* context scenario definition. The resultant generated data for each sensor is shown in Figure 5. This generated multiple sensor stream can be processed by a context interpreter which in turn extracts context information for use by a context aware application.

5.1. *User effort and time required for modelling context scenarios*

The length of time needed for modelling a context scenario is initially long (can span days) if no previously defined sensors, sensor data generation behaviours, context definition and context scenario definitions exist. Defining a sensor data generation behaviour is also a time consuming processes as it involves the composition of a piecewise mathematical function capable of generating a desired sensor stream. Once a repository of sensor definitions and sensor data generation behaviours and context definitions is created, the time required for composing context scenarios is significantly shortened.

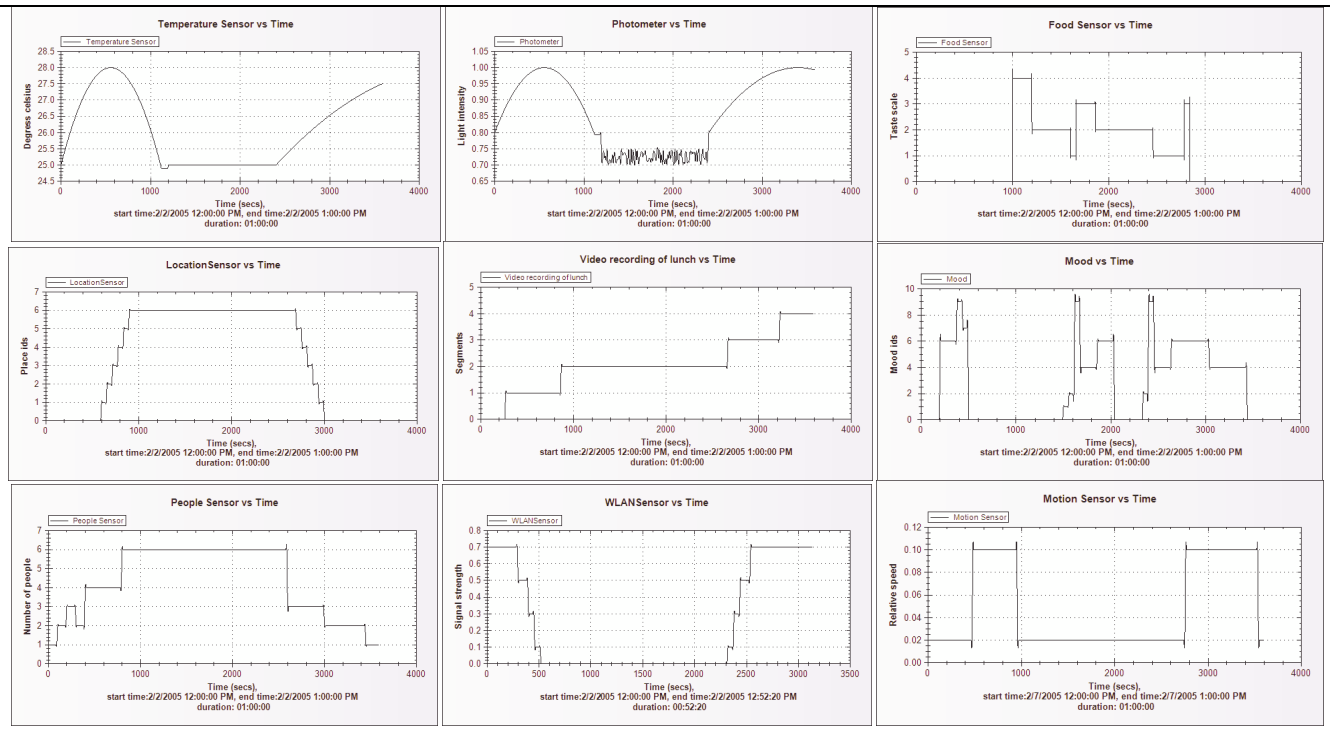


Figure 5: Multiple sensor data that describes the *lunch hour* context scenario

6. Conclusion

In this paper we introduced the concept of a context aware wearable application prototyping environment and proposed a specification for such a prototyping environment. The advantage of using this approach is that context aware applications can be prototyped without the use of real hardware sensors, which can be difficult to assemble or may not exist at the time of investigation of a particular wearable application. The environment also aids in the process prototyping new context aware wearable applications.

ACKNOWLEDGMENT

The financial assistance from the Andrew Mellon Postgraduate Scholarship towards this research is hereby acknowledged. This research was conducted using facilities made available by the Rhodes University Centre of Excellence (COE) in Distributed Multimedia. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily attributed to Rhodes University or the donor.

ABOUT THE AUTHOR

Melekam Tsegaye is a PhD student at the Department of Computer Science, Rhodes University, Grahamstown, South Africa. He is currently conducting research in the following areas: Wearable, ubiquitous and pervasive computing.

REFERENCES

[Brown, 1998] Triggering Information by Context, P. J. Brown, Personal Technologies vol 2, no 1, pp. 18-27, 1998.

[Barton *et al*, 2001] Barton J. and Vikram Vijayaraghavan, UBIWISE, A Ubiquitous Wireless Infrastructure Simulation Environment, 2001. Web: http://home.comcast.net/~johnjbarton/ubicomp/ur/ubiwise/index.htm. Web ref accessed: 04/2005.

[Dey, 2000] Anind K. Dey, Providing Architectural Support for Building Context-Aware Applications, Anind K. Dey, PhD thesis, College of Computing, Georgia Institute of Technology, December 2000.

[Dey *et al*, 2001] 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 (HCI) Journal, Vol. 16, 2001.

[Gemmell *et al*, 2004] Gemmell J, Williams L, Wood K, Lueder R, Bell G, SenseCam Passive capture and ensuing issues for a personal lifetime store, Proceedings of the the 1st ACM workshop on Continuous archival and retrieval of personal experiences, 2004.

[Gilleade *et al*, 2003] Gilleade K, Sheridan J, Allanson J, Liquid: Designing a Universal Sensor Interface for Ubiquitous Computing, Technical Report (comp-001-2003), Lancaster University, Lancaster, UK, 2003.

[Gu *et al*, 2005] Gu T, Keng Pung H.K., Zhang D.Q., A service oriented middleware for building context aware services, Journal of Network and Computer Applications, Volume 28, Issue 1, January 2005.

[Kawsar *et al*, 2004] Kawsar F, Fujinami K, and Nakajima T., "Prottoy": A Context Aware Application Framework, In Proceedings of 2nd International Symposium on Ubiquitous Computing Systems (UCS 2004), Tokyo, Japan.

[Korkea-aho, 2000] Korkea-aho M, Context-Aware Applications Survey, Department of Computer Science, Helsinki University of Technology, Web: http://www.hut.fi/~mkorkeaa/doc/context-aware.html, 2000. Web ref accessed: 02/2005.

[Mann, 2004] Mann S, Continuous lifelong capture of personal experience with EyeTap, Proceedings of the 1st ACM workshop on Continuous archival and retrieval of personal experiences (CARPE 2004), New York, New York, Oct. 15, 2004, p.1 – 21

[Moran *et al*, 2001] Moran P, Dourish P, Introduction to special issue on context aware computing, Human-Computer Interaction (HCI) Journal, Vol. 16, 2001.

[Pascoe, 1998] Pascoe J., Adding Generic Contextual Capabilities to Wearable Computers, 2nd International Symposium on Wearable Computers, Pittsburgh, Pennsylvania USA, 1998.

[PlaceLab, 2004] PlaceLab, PlaceLab An MIT and TIAX imitative, Web:http://architecture.mit.edu/house_n/web/placelab/PlaceLab.pdf, 2004. Web ref accessed: 02/2005.

[Schilit *et al*, 1994] Schilit B.N., Adams N, Want R, Context-Aware Computing Applications, IEEE Workshop on Mobile Computing and Applications, 1994.

[Schmidt *et al*, 2001] Schmidt A., Beigl M, and Gellersen H, There is more to context than location, IMC98-Interactive Applications for Mobile Computing.

[Patel *et al*, 2004] Patel N, Abowd G.D, The ContextCam: Automated Point of Capture Video Annotation, The Sixth international conference on ubiquitous computing, ubicomp 2004.

[Toko, 1998] Toko K, Electronic tongue, Biosensors and Bioelectronics, Volume 13, Issue 6, September 1998.

[W3C, 2001], The Semantic Web, http://www.w3.org/2001/sw/, 2001. Web ref accessed: 02/2005.