Context awareness in mobile computing: A review

Authors:
George W. Musumba[1]
Henry O. Nyongesa[2]

Affiliations:
[1]Department of Computer Science, Dedan Kimathi University of Technology, Kenya

[2]Department of Computer Science, University of the Western Cape, South Africa

Correspondence to:
George Musumba

Email:
musumbaggw@gmail.com

Postal address:
PO Box 657-10100, Nyeri, Kenya

Dates:
Received: 17 Jan. 2013
Accepted: 11 Apr. 2013
Published: 23 May 2013

How to cite this article:
Musumba GW, Nyongesa HO. Context awareness in mobile computing: A review. Int J Machine Learn Appl. 2013;2(1), Art. #5, 10 pages. http://dx.doi.org/10.4102/ijmla.v2i1.5

Context awareness is increasingly gaining applicability in interactive ubiquitous mobile computing systems. In order to apply this concept in design of applications for dynamic environments, it is necessary to understand what constitutes context awareness, the classification of context and context-aware development frameworks. The concept of context awareness is discussed, various approaches to context awareness are reviewed, and important aspects of context-aware mobile computing are analysed. Recommendations for better understanding of context awareness and design principles for development of context-aware applications are provided.

Introduction

Context-aware computing[1] is a mobile computing paradigm in which applications can discover and take advantage of contextual information such as user location, time of day, neighbouring users and devices, and user activity. Many researchers have studied this phenomenon and built diverse context-aware applications.[2] Chen and Kotz[1] examined context-aware systems and applications, types of context used and models of context information, systems that support collecting and disseminating context and applications that adapt to changing context.

Users of computing devices today are faced with diverse devices (mobile or fixed) featuring diverse interfaces and used in diverse environments. This is a step towards realisation of a ubiquitous computing paradigm, or the 'third wave of computing', where specialised devices outnumber users.[3] However, many important pieces necessary to achieve this ubiquitous computing vision are not yet in place.

Most notably, interaction paradigms with today's devices fail to account for major differences between the static desktop and mobile interaction models. Computing devices are now often used in changing environments, yet do not adapt to those changes very well. Although moving away from the desktop model brings a variety of new situations in which an application may be used, computing devices are seldom aware of their surrounding environments. It has been suggested that enabling devices and applications to automatically adapt to changes in surrounding physical and operational environments can lead to enhancement of user experience. Thus information in the physical and operational environments of mobile devices creates a context for interaction between users and devices.

Dey[4] defines context as any information characterising a situation related to the interaction between users, applications and the surrounding environment. Growing research activity within the realm of ubiquitous computing deals with the challenges of context awareness.[2] Although the notion of context can entail very subtle and high-level interpretations of a situation, much of the effort within the ubiquitous computing community takes a bottom-up approach to context. The focus is mainly on understanding and handling context that can be sensed automatically in the physical environment and treated as implicit input to positively affect behaviour of an application.

A different view was introduced by Dey,[4] who proposed use of conceptual models and tools to support rapid development of context-aware applications that could better inform empirical investigation of interaction design and social implications of context-aware computing. This work attempted to enable a new phase of context-aware application development with the intention of helping applications developers understand what context is and what it can be used for, and to provide concepts and practical support for software design and construction of context-aware applications.

Different perspectives on how mobile applications can take advantage of context have been advanced.[5] Thus applications can automatically adapt their behaviour according to discovered

context (active context), or present the context to the user on the fly and/or store it for the user to retrieve later (passive context). This has led to context-aware computing, defined in two ways: firstly, active context awareness automatically adapts to discovered context by changing the application's behaviour; and secondly, passive context awareness presents the new or updated context to an interested user or makes the context persist for the user to retrieve later.

Active context-aware computing leads to new applications, especially in mobile computing, and requires more infrastructure support to help eliminate unnecessary user cooperation and make the technology as 'calm' as possible.[1]

Illustrating this concept, Schmidt[6] provides an example of designing a user interface for a wristwatch. The watch is used both indoors and outdoors, in the dark as well as in sunlight, when running to catch a bus or when attending a boring lecture. A good user interface designer will create varied user interfaces for each situation. The context-aware computing approach enables one to create a context-aware watch, where all situation-optimised designs are combined in a single design. The watch is designed so that it can recognise each of the situations, and then reconfigure itself based on the recognised context. Figure 1 shows a design sketch for a context-aware watch.

Understanding context

A number of studies have investigated the concept of context. Schilit and Theimer[7] refer to context as comprising location, identities of neighbouring users and objects and changes to those objects. Brown, Bovey and Chen[8] define context as location, identities of neighbouring users, time, and environment characteristics such as season and temperature. Ryan, Pascoe and Morse[9] define context as the user's location, environment, identity, and the time. Dey[10] states that context is the user's emotional state, focus of attention, location and orientation, date and time, objects and people in the user's environment. All of these definitions characterise context by examples, and as such their application is difficult.

Schilit, Adams and Want[11] argue that the only important aspects of context are user location, the user's neighbour, and resources near the user. Furthermore, they define context to be subject to the constantly changing execution environment. The environment is thus three-fold:

- computing environment, such as available processors, devices accessible for user input and output, network capacity, connectivity, and cost of computing;
- user environment, such as location, collection of nearby people, and social context;
- physical environment, such as temperature, lighting and noise levels.

Context-aware applications based on these environments are discussed later.

Context information acquisition

The goal of context information acquisition should be to determine what a user is trying to accomplish. Because the user's objective is difficult to determine directly, context cues are used to help infer this information and inform an application on how best to support the user. Context awareness represents a generalised model of input (both implicit and explicit), allowing almost any application to be considered more or less context aware insofar as it reacts to input and the environment.

However, there is divergent opinion as to whether context should only comprise automatically acquired information or also include manually acquired information. In an ideal setting context would be obtained automatically and there would be no need for manual acquisition. However, in the real world not all context information can be sensed automatically and applications must rely on the user to provide it manually.

FIGURE 1: Design sketches that illustrate time visualisations in different contexts. (a) For users running to catch a bus, making it easy to see the minutes in large fonts. (b) For boring lectures and meetings, showing a countdown to the end, with some information to engage the user. (c) Visualisation giving only a very coarse idea of the time, similar to information you get from the sun, to use for example when hanging out with friends – when time does not matter.

Dey[4] observed that designers lack conceptual tools and methods to account for the nature of context and context awareness. As a result the choice of context information used in applications is very often driven by context-acquisition mechanisms available, typically hardware and software sensors. This entails a number of challenges: the choice of sensors may not be most appropriate or optimal; thus any shortcomings of sensors may be propagated up to application level and hinder flexibility of the interaction and further evolution of the application.

Context-aware applications are often distributed because they acquire context information from a number of different sources.[10] As much as the models for application distribution are well known, they are not always appropriate for distributed context information acquisition. Indeed, context awareness is most relevant when the environment is highly dynamic, such as when the user is mobile.

Thus context-aware applications can be implemented on very diverse kinds of computing platforms, ranging from handheld devices to wearable computers to custom-built embedded systems.[12] As a result context-aware applications require lightweight, portable and interoperable systems that can be implemented across a wide range of platforms.

Some of the approaches for acquiring context information are outlined below.

Direct sensing

This is often used in applications with in-built local sensors. The client software gathers the desired information directly from these sensors, without an additional layer for gaining and processing data. Drivers for the sensors are hardwired into the application.[1]

Context server

Multiple clients are permitted access to remote data sources. This is a distributed approach that extends the middleware-based architecture by introducing an access management component with sensor data gathering function moved to the so-called context server to facilitate concurrent multiple access. Winograd[13] describes three different context management models for coordinating multiple processes and components: widgets, networked services and blackboard models.

Classification of context

Dey[14] defined context as 'Any information that can be used to characterise the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves.' Using Dey's[14] definition, Chihani and collleagues[15] observed that context information may be classified according to the described entity.

The context of users may be a combination of various entities such as their identity, activity, location and mood; their social context may be the nature of their relationship with other persons (e.g. family member, colleague, friend); and their physical context might include (for instance) the lighting level of the location where they are. The context of a network may be its quality of service parameters, like round-trip time, and the context of a device may be its capabilities, display features or battery level.[15]

Dey[4] introduced a simple classification of context information based on the entities for which context is assessed, such those including places, people and objects, with four essential characteristics of context information: identity, location, status (or activity) and time. Identity refers to the ability to assign a unique identifier to an entity. Location is expanded to include orientation and elevation, as well as all information that can be used to deduce spatial relationships between entities, such as co-location, proximity, or containment. Status (or activity) identifies intrinsic characteristics of the entity that can be sensed. Time helps characterise a situation, enabling us to leverage the richness and value of historical information.

Application of context

Nowadays ubiquity is fully embedded, with smart devices integrating intelligence for processing various kinds of data. In such an environment the interaction and management of the various devices that a user may hold is a tough task.[14] Context-aware systems are an emerging solution; they can be in charge of supervising the way users interact with the ubiquitous environment for automating users' repetitive actions. For example, a context-aware system can detect that a user never responds to phone calls whilst driving, and thus propose automatically to transfer all incoming calls to the user's voicemail when they are driving.

Dey[4] propose three basic functions that should be implemented by any context-aware application: presentation of information and services, automatic execution of services and storage (and retrieval) of context information. Presentation of information and services refers to functions that either present context information to the user, or use context to propose appropriate selections of actions to the user. Examples here are showing a user their location on a map and possibly indicating nearby sites of interest, presenting a choice of services close by;[11] sensing and presenting input/output information for a group of users;[16] and providing remote awareness of others.[17]

The second function, automatic execution of services, describes functions that trigger a command or reconfigure the system on behalf of the user according to context changes. Examples include a system where a user's desktop environment follows them as they move from workstation to workstation; car navigation systems that recompute driving directions when the user misses a turn; and a camera that captures an image when the user is startled as sensed by biometric sensors.

In the third type of function, storage and retrieval of context information, applications tag captured data with relevant context information. For example, a zoology application may tag notes taken by the user with the location and time of a species observation; and a meeting capture system may provide an interface to access meeting notes based on who was there, when the meeting occurred and where it was located.[4]

Design principles

In this section we describe basic design principles of context-aware applications and depict different models for representing, storing and communicating contextual information. Context-aware systems can be implemented in many ways. The typical approach considers a number of special requirements and conditions, such as location of sensors (local or remote), number of possible users, available resources (such as high-end personal computers or small mobile devices), and extensibility of the system.

Context-awareness models

A context-awareness model is needed to define and store context information in a machine- readable form. Strang and Linnhoff-Popien[18] summarised the most relevant context-modelling approaches based on data structures used for representing and exchanging contextual information in their respective systems. These are highlighted below.

Key-value models

These represent the simplest data structure for context modelling. They are frequently used in various service frameworks, where key-value pairs are used to describe the capabilities of a service. Service discovery is then applied with matching algorithms which use these key-value pairs.

Mark-up models

These use a hierarchical data structure comprising mark-up tags, attributes and content to create profiles which represent a typical mark-up scheme model.

Graphical models

A number of approaches have been proposed where contextual aspects are modelled using Unified Modelling Language.[19]

Object-oriented models

Modelling context using object-oriented techniques offers the full power of object orientation (e.g. encapsulation, reusability and inheritance). Existing approaches use various objects to represent different context information (such as temperature, location, etc.), and encapsulate details of context processing and representation. Access to the context and context-processing logic is provided by well-defined interfaces like the hydrogen model.[20]

Logic-based models

These models have a high degree of formality, and typically facts, expressions and rules are used to define a context model. A logic-based system is used to manage the aforementioned terms and allows addition, updating or removal of new facts. The inference (also called reasoning) process is used to derive new facts based on existing rules in the systems. Contextual information is then represented in a formal way as facts.[21]

Ontology-based models

Ontology represents a description of concepts and their relationships. These models are very promising for modelling contextual information due to their high and formal expressiveness and possibilities for applying ontology reasoning techniques.[18]

User-context perception model

This is a model created to help the designer understand the challenge(s) faced in creating context-aware systems. As an example, a car navigation system works very well if one is in a new city; however, when using it around a familiar area one may sometimes be surprised at the route it tries to direct one to.

To explain this phenomenon with this model, we assume the context-aware system (right side in Figure 2) is of equal quality in both locations; this means that the difference must be on the user's side. The sensory perception (e.g. visual matching of buildings and places you know, based on your sight) is different in familiar and new places. In the new place you lack reference points, and the memory and experience parts in the model differ significantly. In the familiar environment you will have expectations about which route to take and which would be a good choice. In the unfamiliar environment you lack experience and reference points, and hence your expectation is simply that the system will guide you to your destination. The result is that a navigation system that successfully guides you to your destination with a non-optimal route will satisfy your expectations in an unfamiliar environment, but be frowned upon in a familiar environment. In the familiar environment we have a substantial awareness mismatch, whereas when navigating in new surroundings we have minimal awareness mismatch.

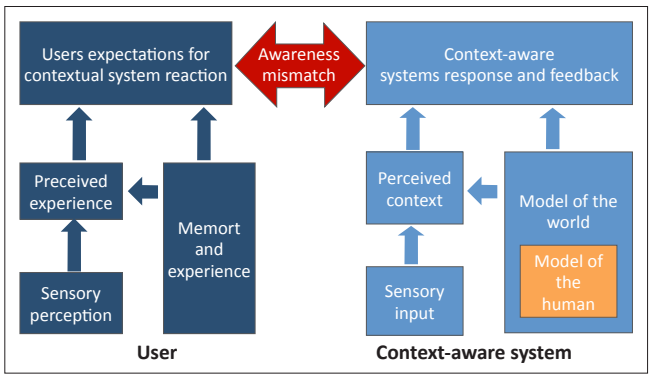

FIGURE 2: The user context perception model.

The quality of context-aware systems, as perceived by the user, is directly related to the awareness mismatch, and a good design aims at designing systems with minimal awareness mismatch. A prerequisite for creating a minimal awareness mismatch is that the user understands what factors have an influence on the system. In the example of a simplistic car navigation system, this factor is only current location and nothing else. In such a case the user knows the system's reactions are based purely on current location and destination, and the user may attribute the system's response to these factors.

In cases where further parameters play a role (e.g. a navigation system that takes traffic into account) it may be more difficult for the user to understand the causalities behind the system's behaviour. The navigation system may suggest different routes in the morning and evening as the traffic situation is not the same. If the user has no knowledge that the system makes routing suggestions based on current location and traffic situation, it is likely that he or she will have a hard time understanding what the system does. As an important rule in the design of context-aware systems, the user should be made aware of the sensory information that the system uses.

Context-aware development frameworks

Schilit[22] presented system architecture to support context-aware mobile computing. The architecture was aimed at supporting gathering of context information about devices and users. Dey, Abowd and Wood[4] identified important features of context and context-awareness and some of the difficulties in building context-aware applications. Three main agent-based components were proposed: device agents that maintain the status and capabilities of devices; user agents that maintain user preferences; and active maps that maintain location information of devices and users.

The proposed architecture did not support or provide guidelines for acquisition of context. Instead device and user agents were built on an individual basis, tailored to the set of sensors that each used. This architecture also did not support context interpretation or storage, leaving different applications to implement these features differently.

CyberDesk

CyberDesk[23] is a framework that was developed to automatically integrate web-based services based on virtual contexts. The virtual context represented information the user was interacting with, such as email addresses, which could be used to display a list of relevant web-based services. Whilst it was limited in the type of context information it could handle, CyberDesk[23] contained many of the mechanisms necessary for a general context-aware architecture. Applications simply specified what context types they were interested in, and were notified when those were available. The modular architecture supported automatic interpretation of individual and multiple pieces of context to produce new sets of context. The architecture also supported abstraction of context information and aggregation thereof. However, it does not support multiple simultaneous applications, querying or storage of context.

Context- and Location-aware Information Service (CALAIS)

CALAIS[24] is another architecture designed to support context-aware applications, and was proposed to solve two problems: the ad hoc nature of sensors and lack of a fine-grained location information management system. An abstraction was developed to hide the details of sensors from context-aware applications. However, similar to Schilit's[11] architecture, there is very little support to aid developers in adding new sensors. Also, the architecture does not support storage or interpretation of context, leaving application developers to provide their own individual mechanisms.

CALAIS supports use of distributed context sensing and provides query and notification mechanisms. An interesting feature is the use of composite events – being able to subscribe to a combination of events. For example, an application can request to be notified when event B occurs after event A with no intervening events. This is a powerful mechanism that makes acquisition and analysis of context easier for application developers.

Context-aware learning spaces (CALS)

CALS[25] is a framework in which context information acquisition typically takes place in informal settings and builds on the foundations of mobile learning. A typical CALS comprises a number of mobile devices (clients), wireless connectivity, a server, and a set of context-aware technologies. Due to challenges in mobile learning such as not accounting for the richness of the environment, context-aware learning integrates the surrounding contextual resources into the virtual learning framework. However, due to the sensitivity of learning context the learner is encouraged to make observations and interact with surrounding objects and phenomena. The flexible nature of CALS allows for creation of new types of applications with minimal development efforts.

A sample layered framework

From the functional viewpoint context-aware systems can be represented as a layered framework (Figure 3), composed from bottom to top by sensors, raw data retrieval, preprocessing, storage or management, and an application layer. A context management system is responsible for retrieving raw data from sensors, abstracting and combining the sensed data into high-level context, and then making it available for the context-aware applications.

The first layer (sensors) is a collection of sensors responsible for raw data from the user environment (e.g. user device, social network, or user access network). The second layer (raw data retrieval) makes use of specific application

programming interfaces or protocols to request data from the sensor layer. These queries must as far as possible be implemented in a generic way, making it possible to replace sensors (e.g. replacing a radio-frequency identification system with a geographical positioning system [GPS]).

The third layer (preprocessing) is responsible for reasoning and interpreting contextual information. It transforms the information returned by the underlying layer to a higher abstraction level (e.g. it transforms a GPS position to one such as at home or at work). Not only sensed or deduced data have to be modelled, but also data describing them (e.g. accuracy and recall, or lifecycle information).

The fourth layer (storage and management) organises the gathered data and makes them available to third parties' applications in a synchronous or asynchronous way. In the first mode the third-party applications use remote method calls to poll the server for changes. In the second mode they subscribe to specific events of interest and are notified when the event occurs (e.g. by a call back).

The fifth layer (application) is where the reactions to context changes are implemented (e.g. displaying text in a higher colour contrast if illumination is bad).

Figure 4 below, adapted from Bardram and Hansen,[26] shows sample reference architecture for context-aware computing systems. In the architecture sensors provide data about activities and events in the real world; perception algorithms make sense of these stimuli and classify the situations into context. Based on the observed context, actions of the system are triggered.

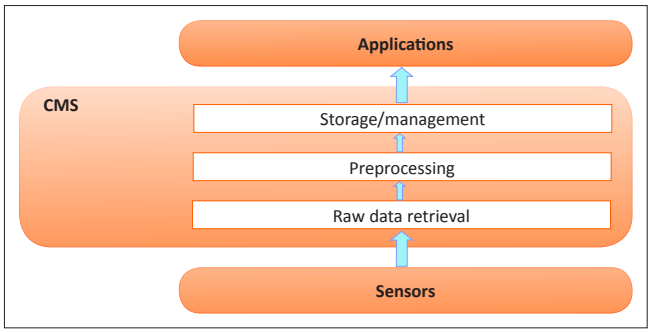

FIGURE 3: Layered framework for context-aware systems.

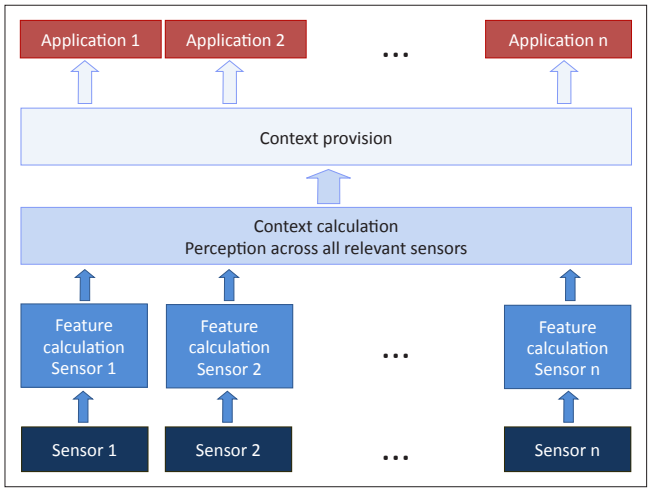

FIGURE 4: A reference architecture for context-aware computing systems.

Context-aware applications

In this section we describe different context-aware systems.

Context-aware browser (CAB)

According to Coppola et al.[27] CAB does the following with varying contexts. On a typical day a user wakes up at say 06:00; this context (of waking up) causes his mobile phone to download and execute wake-up content, through which the user can manage the dwelling's lighting, television and stereo, heater, and so on. As the user has breakfast, CAB provides more functionality to support the user's daily routine; depending on the user's preferences it presents web content that includes the news of the day and weather forecast (see Figure 5a).

FIGURE 5: CAB interface showing generic content for different contexts: (a) as the user has breakfast, CAB presents the day's news and weather; (b) on the way to work, CAB downloads and presents information about the route; (c) on the highway CAB informs the user of an upcoming rest stop and services available.

Next the user enters her car to drive to work. CAB detects the new information from the external environment (e.g. the presence of the Bluetooth installed in the car) and downloads new specific web content that provides information about the car and lets the user personalise and adjust the car's settings. The user drives along a highway. When she nears a tollbooth CAB determines the new context and retrieves the relevant web content (Figure 5b), providing information about the specific highway route. As the user drives, CAB detects the presence of a highway stop (in this case a rest stop with fuel and food services) and retrieves web content informing the user of the estimated distance to the stop and services available (Figure 5c).

Context-based workplace awareness

In a hospital environment with many activities of varying importance and urgency and personnel with diverse skills and responsibilities that need coordination, Bardram and Hansen[26] developed a context-based workplace awareness system to effectively achieve this objective. Their AWARE architecture has both AwarePhone and AwareMedia. The AwarePhone is an application that runs on a smart phone, which can display a contact list for a user and support simple text messaging. The working context for each user is displayed on the contact list, enabling the user to maintain awareness of the context of a colleague before initiating a call. The AwarePhone is integrated with telephone functionality on the phone as well as the messaging system in the AWARE architecture.

Figure 6 shows how users are displayed in a contact list on the AwarePhone. This phone belongs to 'Anette Row' and she can see the social context of her contacts. For example, in Figure 6 the surgeon 'Jens Ole Storm' has a status labelled 'opr'(i.e. operating); he is scheduled to perform an operation called 'Ucementeret Hofte' (a hip replacement operation), and is located in 'OR4', for example operating room 4. It would therefore be unwise to try and call him right now.

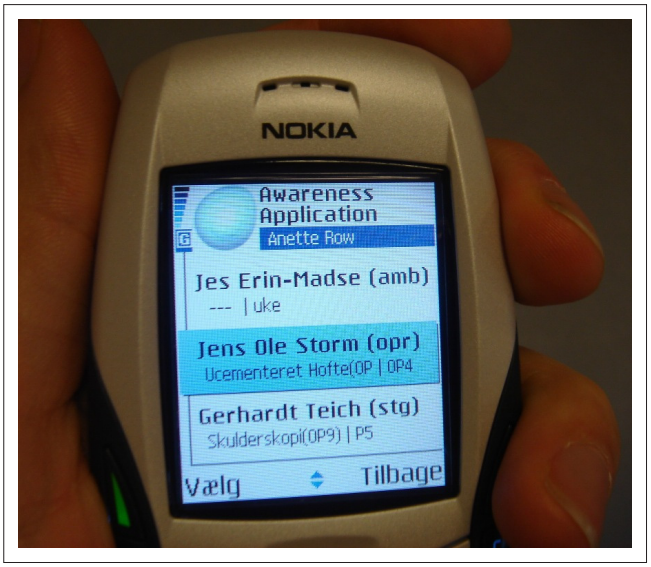

FIGURE 6: Users listed on the AwarePhone.

The AwareMedia system is designed as a large, publicly accessible interactive surface that allows clinicians to maintain a shared, temporal, spatial, peripheral awareness of the progress of work in an operating room or suite. Resembling the whiteboards used pervasively in hospitals, AwareMedia supports publicly available interactive displays mounted on walls in relevant places like the coordination centre and operating room.

ConaMSN: A context-aware messenger

Hong, Yang and Cho[28] created a context-aware messenger (ConaMSN), a system that could recognise various users' contexts, such as level of stress, type of emotion and activity, based on information collected from wearable sensors, and share the context through a messenger application. From a user's physiological information and movement collected using an armband and accelerometers, ConaMSN infers various contexts with modular dynamic Bayesian networks, and visualises them with a set of icons.

By exchanging contextual information of users, ConaMSN lets them know their friends' situation and improves electronic communication. For example, users can give words of encouragement to friends who are feeling gloomy, and be told when to expect a slow response since a friend is involved in some activity.

InCarMusic: Context-aware music recommendations

Context-aware recommender systems (CARS) have been gaining more attention, and various techniques have been introduced to improve their performance.[29] InCarMusic is a mobile application (Android) that offers music recommendations to car passengers after they have entered ratings for items using a web application. If the user did not previously enter any ratings, then the recommendations are adapted solely to the contextual situation and not to the user's long-term preferences.

HEP: Context-aware communication service provision

Information and communication technology advancements in professional environments have enhanced communication between coworkers, although adding a certain amount of stress as they lose control over the way they can be reached and when. Also, the diversity of the communication tools used (e.g. email, instant messaging, video conferencing) amplifies the amount of notifications or interruptions. This may cause degradation of worker performance in his current activity or influence choice of future activities.[30]

To mitigate the workers' performance degradation, Chihani et al.[15] developed HEP, a context-aware system for recommending communication means for enterprise employees. The system publishes real-time information describing their status, as illustrated in Figure 7, emotions, activities and workload. The published information is the

result of processing diverse input streams concerning usage of communication services (phone, instant messaging, email, calendar).

A status corresponds to level of availability of a user on a given communication service (e.g. email, instant messaging, phone). Such information is used by the caller to decide if he can interrupt the callee, and if it is better to use a communication service (e.g. email) than another service (e.g. phone) in order to reach them. For instance, say Alice wants to call Bob on an urgent matter. Bob is on a conference call, but is still reading his emails and answering them. With HEP Alice will see that Bob is busy on the phone but available by email. She thus decides to send him an email instead of calling him, although her need is urgent.

Adaptive and context-aware user interfaces

Context-aware user interfaces are a special case of context-aware functions. A very simple example of a context-aware user interface is the backlight of a device that is switched on when the environment is dark. Further examples are audio profiles that suit a particular situation or screen layouts optimised for a given context. On a mobile device the input modalities may be dependent on the context (e.g. in a car a mobile device may use a simple menu with a large font that can be activated by simple voice commands, but the same device may present a more complex user interface when used during a meeting, as seen on the iPad's differing orientations, shown in Figure 8).

Conference Assistant

The Conference Assistant examines the conference schedule, topics of presentations, user's location and research interests to suggest which presentations to attend. Whenever the user enters a presentation room, Conference Assistant automatically displays the name of the presenter, title of the presentation and other related information. Available audio and video equipment automatically record the slides of the current presentation, comments and questions for later retrieval.

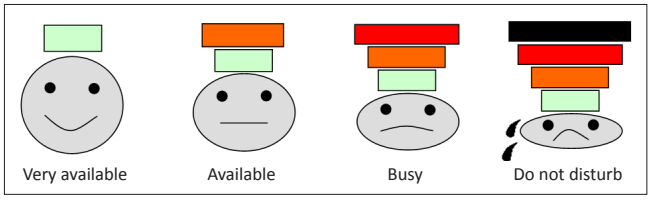

FIGURE 7: HEP statuses.

FIGURE 8: An iPad switching orientation of the screen.

Office Assistant

This is an agent-based application that interacts with visitors at the office door and manages the office holder's schedule. The assistant is activated when a visitor approaches, detected by pressure-sensitive mats placed on both sides of the office door. It adapts its behaviour to contextual information such as identity of the visitor, office owner's schedule status and the owner's willingness to see the visitor.[31]

Other context-aware applications surveyed by Chen and Kotz[1] include Teleporting, a tool to dynamically map the user interface onto the resources of the surrounding computer and communication facilities; Shopping Assistant, an application designed to guide shoppers through a store, provide details of items, help locate items, point out items on sale, and do a comparative price analysis; and adaptive global systems for mobile phone and personal digital assistants, where profiles of the mobile phone are selected automatically based on recognised context. The phone chooses to ring, vibrate, adjust ring volume or keep silent, depending on whether the phone is in hand, on a table, in a suitcase, or outside. Another application is Location-aware Information Delivery, where each reminder message is created with a location; when the intended recipient arrives at that location the message is delivered via voice synthesis without requiring the user to hold the device and read it on screen.[32]

Security and privacy issues

It is important to address security and privacy issues in context-aware mobile computing. There are two key security concerns with context-aware systems:[33] firstly, ensuring privacy of location and identity information, and secondly, ensuring secure communications. Authenticating location information is difficult because common sensor systems typically rely on technologies such as active badges that can be removed from the mobile objects they represent. In addition to securing the content of communication, origin and destination of content should also be protected.

However, few existing context-sensitive systems provide satisfactory security solutions, whilst others choose to ignore security and privacy concerns altogether. Optimal privacy guarantees are in general difficult and expensive to provide.[33] However, users should be able to have control over their contextual information and who may access to it. Thus system architectures need to provide user-controllable trade-offs between privacy guarantees and functionality. Yet it is difficult to be specific about what context information should be visible to who, and when.

Baldauf and colleagues[2] introduced the concept of context ownership. Users are assigned to sensed context data as their respective owners, and allowed to control other users'

access to it. New components involved in this access control are mediated widgets, owner permissions, base objects and authenticators. The mediated widget is an extension of a basic widget which contains a so-called widget developer specifying who owns the data being sensed. Ultimately the problem is that whilst any technology is rarely in itself inherently bad, it can be used for good or bad purposes.[34]

Future directions

With devices enabling us to monitor users in more detail, context awareness will be included in consumer devices to an ever-increasing degree. Imagine if technologies like cameras and the Kinect (a motion-sensing input device by Microsoft for the Xbox 360 video game console) were included in appliances, devices and your office and home environment. Recognising where people are and what they do will enable designers to create attentive applications that look at what you do and react appropriately. The shower will recognise which member of your family is going to use it (e.g. based on body profile) and preselect that person's favourite temperature.

Designers may explore how appliances can be operated with minimal interaction; potentially just 'being there' is enough to work with your environment. Here a central challenge is to provide ways in the user interface to correct wrong choices made by the system, so that the user feels in control.

According to Bulling and colleagues[35] a rich source of information on context that has not been used yet is the movement of the eye. The dynamics of eye movements as we engage in specific activities reveal much about these activities (e.g. reading). Similarly, specific environments or locations influence our eye movements (e.g. driving a car). Finally, eye movements are strongly related to the cognitive processes of visual perception, such as attention, visual memory or learning. In addition to physical activity or location, eye movement analysis could help us infer these processes in real-world settings. Eventually this might let us extend the current notion of context with a cognitive dimension, leading to cognition-aware systems that enable novel types of user interaction not possible today.

There is a need to extend Kiefer, Straub and Raubal's[36] discussions on impact of location as context in mobile eye-tracking studies that extend to large-scale spaces, as compared to the static eye tracking and variable contexts that characterise natural vision.[37]

It is exciting to think about how rich sensing and communication will change the way we live. Together with Kristian Kersting and Marc Langheinrich, Schmidt wrote an article entitled 'Perception beyond the here and now'[38] which discusses how sensor-equipped computing devices are overcoming longstanding temporal and spatial boundaries to human perception.

Conclusion

This paper surveys seminal literature to better understand context awareness in mobile computing. It explores and discusses how context is represented, classified and applied, and also discusses design principles with regard to context-awareness models. It also examines existing context-aware systems and frameworks supporting development of context-aware applications, and security and privacy issues with future developments in the research area.

Major milestones have been achieved in the understanding, categorisation and development of context-aware systems. It can be concluded that context awareness is a key factor in development of new applications for ubiquitous mobile computing. A prevailing concern, however, is the issue of security and privacy.

Some of the major issues that stand out from this review are the representation, categorisation and acquisition of context information. Different researchers have approached these issues from individual perspectives in proposing frameworks to describe and handle context. Standardised protocols and formats are important for development of context-aware systems, including unified communication and security protocols.

In conclusion, considering all the issues and aspects of context awareness highlighted here, more understanding of requirements in design and development of context-aware applications is necessary. Our research investigates application of context awareness in modelling of virtual enterprises.

Competing interests

The authors declare that they have no financial or personal relationship(s) which may have inappropriately influenced them in writing this article.

Authors' contributions

G.W.M. (University of Technology, Kenya) was responsible for the review of the articles, problem definition and writing of the article. He is also the main correspondent. H.O.N. (University of the Western Cape) was responsible for guiding and supervising the research.

References

1. Chen G , Kotz D. A survey of context-aware mobile computing research. Dartmouth: Dartmouth Computer Science Technical Report; 2004.
2. Baldauf M, Dustdar S, Rosenberg F. A survey on context-aware systems. Int J Ad Hoc Ubiquitous Computing. 2007; 2(4):263–277. http://dx.doi.org/10.1504/IJAHUC.2007.014070
3. Weiser M. The computer for the 21st century. Scientific American. 1991;265(3):66–75. http://dx.doi.org/10.1038/scientificamerican0991-94
4. Dey AK. Understanding and using context. Pers Ubiquitous Computing. 2001;5(1):4–7. http://dx.doi.org/10.1007/s007790170019
5. Long S, Kooper R, Abowd GD, Atkeson CG. Rapid prototyping of mobile context-aware applications: The Cyberguide case study. Wireless Networks. 1997;3:421–433. http://dx.doi.org/10.1023/A:1019194325861

6. Schmidt A. Context-aware computing: context-awareness, context-aware user interfaces, and implicit interaction. In: Soegaard M, Dam RF, editors. The Encyclopedia of Human-Computer Interaction, 2nd edition. Aarhus, Denmark: The Interaction Design Foundation; 2013.
7. Schilit B, Theimer M. Disseminating active map information to mobile hosts. IEEE Network. 1994; 8(5):22–32. http://dx.doi.org/10.1109/65.313011
8. Brown PJ, Bovey JD, Chen X. Context-aware applications: from the laboratory to the marketplace. IEEE Pers Comm. 1997; 4(5): 58–64. http://dx.doi.org/10.1109/98.626984
9. Ryan N, Pascoe J, Morse D. Enhanced reality fieldwork: the context-aware archaeological assistant. In: Gaffney V, Leusen MV, Exxon S, editors. Computer Applications in Archaeology. Oxford: British Archaeological Reports; 1997.
10. Dey AK. Context-aware computing: The CyberDesk project. Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments; 1998 March 23–25; Stanford. Menlo Park, CA: AAAI Press, 1998; p.51–54.
11. Schilit B, Adams N, Want R. Context-aware computing applications. Proceedings of the 1st International Workshop on Mobile Computing Systems and Applications, 1994; Los Alamitos, CA. Washington: IEEE Network; 1994, p.85–90.
12. Bauer M, Heiber T, Kortuem, G, Segall Z. A collaborative wearable system with remote sensing. Proceedings of the 2nd International Symposium on Wearable Computers (ISWC'98), 1998 Oct 19–20; Pittsburgh, PA. Washington: IEEE Computer Society; 2000, p.10–17.
13. Winograd T. Architectures for context. *Human-Computer Interaction J.* 2001; 16(2):401–419. http://dx.doi.org/10.1207/S15327051HCI16234_18
14. Dey AK, Abowd GD. Towards a better understanding of context and context-awareness. Proceedings of the Workshop on the What, Who, Where, When and How of Context-Awareness, 2000 April; The Hague, The Netherlands. New York: ACM Press; 2000.
15. Chihani B, Bertin E, Jeanne F, Crespi N. Context-aware systems: a case study. In: DICTAP'11: The International Conference on Digital Information and Communication Technology and its Applications, 2011; Dijon, France. Heidelberg: Springer; 2011, p. 718–732.
16. Salber D, Dey AK, Abowd GD. The Context Toolkit: Aiding the development of context-enabled applications. In: Proceedings of the CHI '99 Conference on Human Factors in Computer Systems. New York: ACM Press, 1999; p.434–441.
17. Schmidt A, Takaluoma A, Mäntyjärvi J. Context-aware telephony over WAP. Pers Technol. 2000; 4(4):225–229. http://dx.doi.org/10.1007/BF02391563
18. Strang T, Linnhoff-Popien C. A Context Modeling Survey. In: First International Workshop on Advanced Context Modeling, Reasoning and Management, held as part of UBICOMP 2004, Nottingham, UK [homepage on the Internet] 2004 [cited 2013 May 2]. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary;jsessionid=C6D750C195B3CB512507B04E3F58D46D?doi=10.1.1.2.2060
19. Sheng QZ, Benatallah B. Context-UML: a UML-based modeling language for model-driven development of context-aware web services. In: Proceedings of the International Conference on Mobile Business (ICMB'05), 2005. New York: IEEE, p.206–212. http://dx.doi.org/10.1109/ICMB.2005.33
20. Hofer T, Schwinger W, Pichler M, Leonhartsberger G, Altmann J. Context-awareness on mobile devices – the hydrogen approach. In: Proceedings of the 36th Annual Hawaii International Conference on System Sciences, Hawaii, 2002 [homepage on the Internet]. 2002 [cited 2103 May 2]. Available from: http://www.citeulike.org/user/mboehmer/article/3502905
21. McCarthy J, Buvac S. Formalizing context (expanded notes). In: Buvac S, Iwahska L, editors Working Papers of the AAAI Fall Symposium on Context in Knowledge Representation and Natural Language, California, 1997; Menlo Park: American Association for Artificial Intelligence, 1997; p.99–135.
22. Schilit WN. System architecture for context-aware mobile computing [unpublished PhD thesis]: Columbia University, 1995.
23. Dey AK, Abowd GD, Wood A. CyberDesk: A framework for providing self-integrating context–aware services. Knowledge Based Systems. 1998; 11(1):3–13. http://dx.doi.org/10.1016/S0950-7051(98)00053-7
24. Nelson G J. Context-aware and location systems [unpublished PhD dissertation]: University of Cambridge, 1998.
25. Teemu HL. Technology Integration in Context-Aware Learning Spaces [unpublished PhD thesis]: University of Eastern Finland; 2011.
26. Bardram JE, Hansen TR. Context-based workplace awareness. Comp Supp Coop Work. 2010; 19(2):105–138. http://dx.doi.org/10.1007/s10606-010-9110-2
27. Coppola P, Della Mea V, Di Gaspero L et al. The context-aware browser. Intelligent Systems IEEE. 2010; 25(1):38–47. http://dx.doi.org/10.1109/MIS.2010.26
28. Hong JH, Yang SI, Cho SB. ConaMSN: A context-aware messenger using dynamic Bayesian networks with wearable sensors. Expert Syst Applic. 2010; 37(6):4680–4686. http://dx.doi.org/10.1016/j.eswa.2009.12.040
29. Baltrunas L, Kaminskas M, Ludwig B et al. Incarmusic: Context-aware music recommendations in a car. E-Commerce Web Technol. 2011; 89–100.
30. Hudson JM, Christensen J, Kellogg WA, Erickson T. I'd be overwhelmed, but it's just one more thing to do: Availability and interruption in research management. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2002 Apr 20–25; Minneapolis, Minnesota. New York: ACM, 2002.
31. Yan H, Selker T. Context-aware office assistant. In: Proceedings of the 2000 International Conference on Intelligent User Interfaces, New Orleans, LA, 2000 Jan 9–12. New York: ACM Press, 2000; p. 276–279.
32. Want R, Hopper A, Falcão V, Gibbons J. The Active Badge location system. ACM Trans Inf Syst. 1992; 10(1):91–102. http://dx.doi.org/10.1145/128756.128759
33. Spreitzer M, Theimer M. Providing location information in a ubiquitous computing environment. In: Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles, 1993 Dec 5–8; Asheville, NC. New York: ACM Press, 1993; p. 270–283. http://dx.doi.org/10.1145/168619.168641
34. Want R, Hopper A, Falcao V, Gibbons J. The Active Badge location system. ACM Trans Inf Syst. 1992; 10(1):91–102. http://dx.doi.org/10.1145/128756.128759
35. Bulling A, Roggen D, Troster G. What's in the eyes for context-awareness? Pervasive Computing, IEEE. 2011; 10(2): 48–57. http://dx.doi.org/10.1109/MPRV.2010.49
36. Kiefer P, Straub F, Raubal M. (2012, March). Towards location-aware mobile eye tracking. In: Proceedings of the Symposium on Eye Tracking Research and Applications; 2012 Mar; Santa Barbara, CA. New York: ACM; 2012, p. 313–316.
37. Franchak JM, Kretch KS, Soska KC, Babcock JS, Adolph KE. Head-mounted eye-tracking of infants' natural interactions: a new method. Paper presented at Symposium on EyeTracking Research and Applications (ETRA 2010); Austin, Texas.
38. Schmidt A, Langheinrich M, Kersting K. Perception beyond the Here and Now. IEEE Computer. 2011; 44(2):86–88. http://dx.doi.org/10.1109/MC.2011.54