Context-aware mobile applications design: implications and challenges for a new industry

Abstract

Context-aware computing is slowly becoming the new mobile paradigm in which applications can discover and use information "out and about". Typical sources of knowledge about context are the device's location, data about the environment at large, the mobile device's prior activity log and even the user's biometrics. The mobile industry agrees that this paradigm improves the appeal and value of applications by personalising and adapting them to the context in which they run. However, capturing contextual information and processing it to enhance or create a new application is a daunting task: it involves scattered systems and infrastructures and an increasingly wide array of heterogeneous data, architectures and technological tools. In this paper, we explore and analyse existing mobile context-aware applications and the proposed frameworks that enable them. The paper aims to clarify the technological choices behind context-aware mobile applications and the challenges that still remain ahead for this area to fulfil the promises it offers.

1. Introduction

New ubiquitous computing applications based on intelligent systems are designed to be aware of the context in which they run. They require special properties that traditional computing applications do not support, such as capturing and understanding real world information and being self-adaptive and proactive [1]. Ever since the concept of context-awareness was identified, it has been the subject of significant academic interest. However, just in the last two years have some large players (e.g., Cisco Systems, Nokia, Samsung, and others) begun to explore its potential for context-enriched services [2].

The initial motivation for context-aware computing was to reduce the explicit information a user needs when interacting with a computer [3]. In general, context-awareness has gained importance in software development, and it is currently used to enhance applications, although still in modest ways. The usual procedure involves adding meaning to the user's input by including "real world" information that can help the system to understand explicit input and make interactions more personal and effective. In fact, we engage in this process of contextualising continuously and unconsciously when interacting with humans or computers. Transforming it into a conscious process with the help of technology can make interactions more effective [4].

The mobile domain seems to be the most suitable arena to test the concept of context-awareness. In mobile systems, changes in the user's environment are fast and unexpected and have significant impacts on the user's needs; reacting to the user's context is the key to a better match with user's expectations. For example, the requirements of interaction with a mobile device might change according to context: if we are in a meeting in a business situation, we might want our cell phone to be silent automatically so that it will not disturb the meeting. On the other hand, if we are in a social environment, we might want our cell phones' volume to be turned up high so that we can hear it in what is probably a noisy environment.

In addition, mobile phones are a desirable computing platform for context-awareness because they are both pervasive and centred around the users' lives. New smartphones are continuously able to capture contextual information such as location, interaction with the mobile device, surrounding information (i.e., video, audio, noise level, temperature). Therefore, a primary enabler for mobile context-awareness adoption is the proliferation of sensors and their inclusion in new smart devices. Sensors capture contextual information from the environment and convert it into signals that can be read and transformed into meta-data. They are now routinely included as features of many portable devices. For instance, in 2009 there were 435.9 million accelerometers, gyroscopes and pressure sensors in mobile devices; this number is expected to rise up to 2,2 billion in 2014 [5]. Mobile devices have also begun to increase their processing power, and, complementarily or alternatively, they have increasing broadband connections to mobile "cloud computing" services [6]. In fact, including context information has sparked an explosion of new mobile application development [7], and many mobile devices already include some simple context functionalities, for instance, functionalities connected with the inclination or movement of the device. Currently, context-aware applications are typically commercialised as mobile augmented reality services, in which information coming from the virtual world (such as the Internet) is superimposed on physical objects and can be browsed with specific software. They are a promising area and are expected to achieve revenues of more than $700 million by 2014 [8].

However, there is still a lack of understanding of context-aware mobile systems from the general development point of view. In particular, this paper reviews the main examples of context-aware mobile applications, whether they are pilots or commercial systems, and analyses their underlying architectures and computational models in an attempt to understand the main challenges facing the industry in taking this concept into the mainstream. For this purpose, the rest of the paper is organised as follows. Section 2 reviews existing examples of context-aware mobile applications through several case studies. In Section 3, we introduce design principles to contextualise mobile applications, analysing the development approaches used in various context-aware systems. The results of this analysis will provide some insight into the implications and challenges confronting context-awareness, which we will analyse in Section 4. Finally, in Section 5, we conclude the paper by summarising the main findings of this work and outline some of the future areas of interest in mobile context-awareness.

2. Context-aware applications

To create context-aware applications, we must first understand the meaning of context. The debate on the precise definition of notion has been going on since the early 1960s, and to date there is no clear consensus [9]. For the purposes of this paper, we define context as any relevant information that can be used to characterise the situation of an entity when interacting with a user at a given point in time[1]. An entity can be a person, a device, a software application, or almost anything else. In our definition, we adopt a data-centric approach because we believe that context derivation is related to data that affect a situation, and, therefore, it must be understood from this point-of-view when

[1] Other authors adopt different points of view in defining context. Some use a process-based definition according to which context refers to the space and circumstances in which an action is taking place 10. Coutaz, J. and J.L. Crowley, *Context is key.* Communication of ACM, 2005. **48**(3): p. 49 - 53. From a technological point of view, it simply refers to the physical world that surrounds the mobile device 11. Gellersen, H.W., A. Schmidt, and M. Beigl, *Multi-Sensor Context-Awareness in Mobile Devices and Smart Artifacts.* Mobile Networks and Applications, 2002. **7**(5): p. 341–351. From a semantic point of view, a context is a "set of attributes and predetermined values, labeled in some meaningful way and associated with desirable semantics". 12. Lassila, O. and D. Khushraj. *Contextualizing Applications via Semantic Middleware*. in *Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05) 2005 IEEE.* 2005.. From a state point of view, context is defined as the facts or circumstances that surround a situation or event 13. Chen, G. and D. Kotz, *Solar: A pervasive-computing infrastructure for context-aware mobile applications*, in *Dartmouth Computer Science Technical Report TR2002-421*. 2002, Dartmouth College.

developing applications that are context-aware. Context derivation is related to the data that affect a situation, and this fact is not taken into account in any of the prior definitions given.

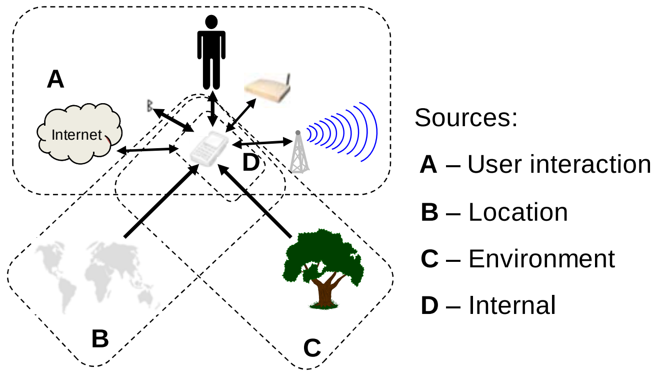

Figure 1. Context data sources categories

Context-aware mobile computing refers to a general class of systems that can sense a continuously changing physical environment. Context data sources can be elements of the entity's mobile system, for example, the status of the connection between the mobile device and the network, actions taken by the entity, texting or making a phone call, or more conventional information about the environment such as location, time of day, or nearby people and devices. Context data are heterogeneous and can be retrieved in a variety of ways, such as through sensors, network information, device status, browsing information, device profile [14] and even asking for information from the "cloud". In addition, applications may have different needs for context information in terms of their levels of abstraction [12].

We categorise context data sources into four main classes (as shown in Figure 1).

A- User interaction with the device, which includes mobile usage activity or biometrics and mobile communications or information coming from internet based services such as web 2.0 or cloud computing.

B- Location data related to what is happening in the mobile device's surroundings or ambient information related to the physical world.

C- Environmental information, such as heat, light, sound levels or speed.

D- Internal device information, such as battery charge level, cost of computing or keypad/display interaction.

To better understand the possibilities of mobile context-awareness, within this section we review some examples of solutions running over different types of mobile devices (PDAs, mobile phones or netbooks) applied to different industries and domains: transportation, health care, automotive, academics, gaming, search engines and social networking, and general usability improvement. The examples we describe are either pilots in the research world or commercial solutions in industry.

When analysing case studies, we focus on the following elements: motivation, frequency and involvement. The motivation of making the application context-aware refers to how contextual information is leveraged to be useful. The frequency of context-awareness is how often contextual information is retrieved and applied: whether it is used based on events or periodically, i.e., in a discrete or continuous mode. The involvement of the user in context-awareness is considered to be active or passive [15]: the user can either request an action based on the context, which is called a pull mode, the context-awareness adaptation can be done actively without the user's approval, a push mode, or there can be an intermediate push-pull mode in which the action is initiated independently but requires a user confirmation.

Case study 1: improving interface usability

Data from mobile phones' or other entities' user interactions and environment are used as inputs to improve their interfaces and usability, shifting among pre-defined profiles with the goal of reducing the need for the user's attention. In all cases, context-awareness is performed in a push mode in which the device or the application adapts itself automatically without awaiting for confirmation from the user. Context data are captured through the mobile device's internal sensors: microphone, camera, tilt, touch and IR. Contextual information is provided constantly during the use of the application, and it automatically senses contextual information periodically.

Examples of pilots are automobile infotainment applications (Sharma, Kuvedu-Libla et al. 2008), context-aware phones (Siewiorek, Smailagic et al. 2003; Schmidt and Laerhoven 2001), adaptive note pad applications (Schmidt 2000) and PDA usability (Hinkley, Pierce et al. 2005).

Case study 2: improving industry solutions

Data from a device's location and user interactions, possibly including biometrics are used as inputs to solutions in specific domains, such as automotive, advertising, academics, games or search engine. In all cases, the user must have some active role, at least launching the application to trigger the context-awareness, which is typically based on environmental and location information. Being context-aware improves the solution by providing specific contextual data that are often key elements in its functionality. Context data are captured through different communications systems and connectivity sensors such as location systems (GPS), short range communications (Bluetooth, etc.), mobile communications (GPRS, UMTS, etc.) and WLAN (WiFi, etc.). Context-awareness is maintained over time, and during utilisation, the application is continuously looking for contextual information. Typically context plays an active role in a push-pull mode in which a prior confirmation from the user is requested.

Examples of pilots are a parking assistant for an automobile (Rasmussen, P. Østergaard et al. 2007), a mobile advertising system (Aalto, Göthlin et al. 2004) and an academic solution [16]. Example of commercial solutions include mobile gaming applications such as the game Alien Revolt by M1nd Corporation and the carrier Oi that links a cell phone's connection to physical spaces and users [17].

Case study 3: improving workplace spaces

Data on location and user interactions are used as inputs to improve workspaces in domains such as construction sites, train stations, hospitals and offices. Context-aware applications store information about the workspace and display it to the user on demand based on the user's location. Being context-aware facilitates the task management while minimising the interaction required from the user. Context data are captured through different connectivity sensors such as IR, WLAN, or RFID. Context-awareness is maintained over time; during the whole (working) day the application receives contextual information and updates its data accordingly. Context plays an active role in push mode; the system automatically incorporates contextual information without asking for validation from the users.

Examples of pilots are workforce management at a construction site and workforce management and information system at a train station [16], health care context aware computing in hospital work (Bardram 2004) and general adaptive workspace applications (Schilit, Adams et al. 1994). .

Case study 4: improving search engines

Data from location, environment, user interactions and bio-parameters are used as inputs to improve the results of search in mobile situations. Context data are captured by sensors embedded in the mobile device that extract information from the surrounding environment and that are able to communicate with other sensors (aka the internet of things). The users' profiles and their behavioural patterns are used to provide more meaningful results in the search process. Context-awareness is activated when a query is performed in the search engine application, and depending on the applications further intervention from the user side (pull mode) can be required, in particular if privacy and/or personal data must be compromised.

Examples of proposed applications and pilots include improving the accuracy of search engine results for places surrounding the users, offering a more valuable and entertaining user experience and automatically interfacing with the resources of the internet of things [18].

Case study 5: improving social networks

Location data are used as inputs to improve the experience of mobile social networks. They allow users to share their whereabouts with the contacts on their network by adding their location to their status or their news feeds. This feature improves the social networking application by adding location information in real time. Users check into places and can share where they are and make comments on the place. Context data is captured by the device's embedded GPS or by the user's input through the keyboard. Context-awareness is activated when the location sharing application is launched, and it requires intervention form the user to comply with privacy rules.

Examples of existing commercial applications and pilots are Facebook Places, Google's Foursquare and Gowalla. All of these applications enhance social networking by adding location information that informs users of where their friends are at any given moment [19]. These tools run over most mobile platforms, including RIM, Android, iPhone, Palm, and others.

3. Infrastructure supporting context-aware applications

As shown in the case studies, context-aware mobile applications integrate information from diverse sources. Therefore, they require the coordinated use of devices provided from several distinct industries (Figure 2). These devices supply data about the context in particular formats, with hugely different levels of quality and reliability, and through particular and diverse interfaces. Adding to the complexity, context-aware applications can run on multiple mobile platforms: laptop computers, mini-computers, mobile computers, tablet computers, game consoles, smartphones and a myriad of new types of upcoming small devices, with widely diverse features even within the same category.

Figure 2. Context-aware application interaction with several industry devices

Thus, it is no wonder that several architecture designs have been proposed to develop context-aware applications. However, all of them share a very simple and basic model (Figure 3) that we can summarise in three main stages:

1. **Data capturing**, i.e., acquiring contextual data through sensors.
2. **Context derivation**, i.e., data modelling, analysis and processing.
3. **Action execution**, i.e., triggering an action based on the properties of the recognised context emanating from the processed data.

The flow of information through these three stages is as follows. First, raw contextual data is captured (acquired) by sensors. User permission may be requested at this stage for privacy reasons. Secondly the raw data are "merged" as they may originate simultaneously in different sources (low-level context data) and assigned a level of quality. The result is processed by the reasoning engine, classifying it into data atoms or meta-data (high-level context data). This engine typically uses classification mechanisms based on artificial intelligence tools to match the data atoms with the features of a number of specific contexts, either pre-defined, set interactively by the user or derived by the engine itself. Third and last, based on the properties of the identified context (i.e., a combination of values, a threshold, a match with a profile), a particular action or a set of actions are triggered. User intervention may be required at this stage to confirm/accept the action or to refine it.

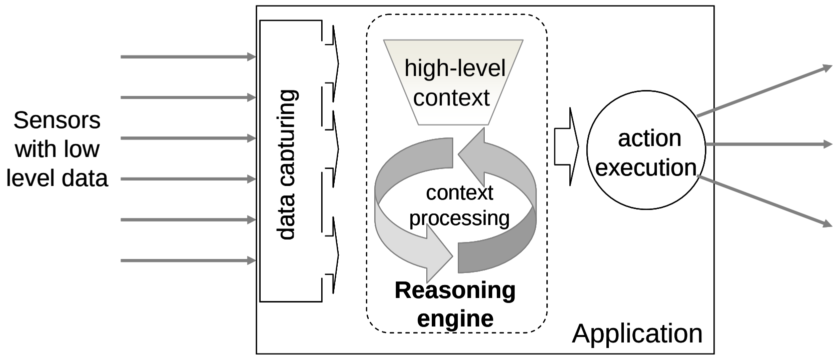

Figure 3. Conceptual architecture of context-aware applications

This basic model is implemented in a specific practical implementation by specifying a design model for the architecture itself and using a set of particular technology tools to implement of the modules that compose the architecture. We explore both elements in detail.

3.1. The architecture modules

Data capturing

This module defines what information should be sensed (captured), which are the available sensors, what information related to the mobile device is relevant to context and where is it stored[2].

There are several possible contextual sources, which can be grouped into 4 areas as explained earlier and shown in Figure 1. Contextual data is captured through sensors, which can be embedded or add-on, internal or external to the device, and physical or virtual. The information can be accessed through APIs[3] that store the sensor's values: name, accuracy, data type, measurement ranges, scale and units. In Table 1, we can see some examples of the map between context data available to mobile devices[4] and the sensor that captures it.

Data	Sensors	Description
Temperature	Thermometer, Cloud	External temperature and/or body heat.
Movement, Angle	Accelerometer, Camera	Inclination, acceleration or motion of the device.
Location	GPS, Camera, other	Position, location and proximity to places.
Tactile interaction	Mouse Joystick Keyboard Haptics	Interaction between the user and the device, indicating its motion (right, left, middle buttons) or the key pressed.
Audio	Microphone	External noise level, type of input (music, speaking) and base frequency.
Light	Camera	Light intensity or spectrum.
Images, Video	Camera	Recognition of surrounding environment and relative position
Time	Clock, time	Time measurement on the precise moment.
Ring	Profile	Phone ring mode, silent, meeting, outdoors or loud.
Connection	Communications module	Indicates which network is the phone connected to a: PAN, MAN, LAN or WAN.
Device usage	Internal Log	Use of the mobile device, information typed, accessed, etc
Social	Cloud	Information about the status of the social network

Table 1. Map between context data and sensors (based on Java API)

Context processing

This module deals with transforming contextual data into a "plain mobile natural language" so that it can be understood and used properly by the application. Therefore, an ontology with the appropriate vocabulary is required. It acts as a semantic

[2] This last part is sometimes called "reality mining", the collection of machine-sensed environmental data pertaining to human social behavior, such as physical proximity of people, phone calls, movement, etc

[3] APIs are application programming interfaces used for development purposes.

[4] Based on the information of the Java API and on the research by 20. Schmidt, A. and K.V. Laerhoven, *"How to Build Smart Appliances?"*. IEEE Personal Communications, 2001. on how to build smart appliances

framework that translates context data into context information[5] based on five principles:

1. The information domain is restricted to the range of meanings emanating from context data sources.
2. The ontology design should be simple so that anyone can understand it.
3. It should be accessible, so that the queries are as straightforward as possible.
4. The design should be as generic as possible so as to support different types of context data and support different types of inference for queries.
5. The design should be flexible so that it can cater for all possible user interests and situations as required.

Each context variable is defined using a group of values (mandatory or not) with some properties that form a context category as a verbal description (see the examples in Table 2).

	Property	Description
Mandatory	Context type	Category of context
	Context type level1	First level of context that gives specific meaning to the context category.
	Context type level2	Second level of context that gives specific meaning to the context category type.
	Value	Raw numerical value that context variable takes (type + subtype).
	Timestamp	Latest time the context value was retrieved.
Optional	Probability	Degree of quality/confidence the context value has.
	Source	Where does the information come from, where is it located.
	Attributes	Any other information that might be useful.

Table 2. Context ontology variables (research sources adapted by the authors)

Reasoning engine

This module comprises the mechanisms used to pre-process and normalise contextual data for the interpretation of context, which usually derived from machine learning algorithms. They are an intersection of computer science, statistics and cognitive[6] science. They are designed to learn to predict output from an input pattern; either in an automatic way, with no human intervention in the learning process, or in an inductive way, learning from a given observed sequence of events.

Although data are usually unstructured, there is always some potential relationship ("correlation") between data that can be found through these algorithms (see Figure 4). Examples of commonly used techniques for data modelling and prediction are Bayesian algorithms and neural networks[7].

5 Ontologies provide a way of categorizing information and giving a meaning to it, while vocabularies store these definitions. Together they form a semantic framework that provides the meaning of the data.

6 Cognitive technologies are used in a loose sense to "understand" user behaviour, user intentions and personal context.

7 Bayesian networks are probabilistic graphical models that represent a set of random variables and their conditional dependences via a graph. Neural networks mimic the properties of biological neurons, which fire or stay at rest given an input.

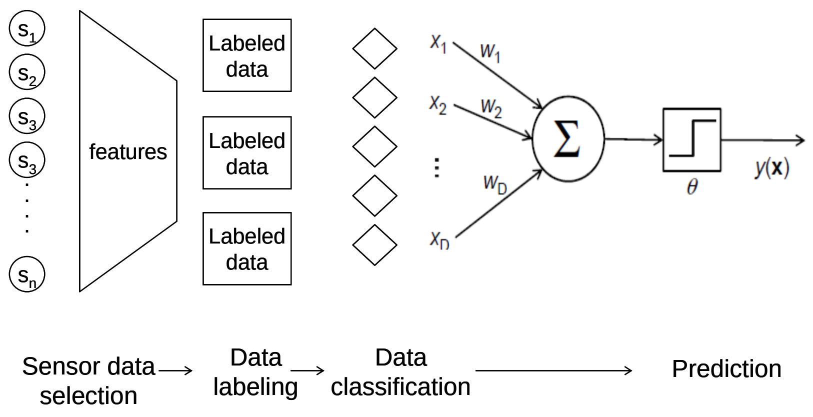

Figure 4. Data reasoning process

Context data are labelled based on the degree of uncertainty. Each context atom is associated with a conditional probability related to the outputs [21]. The system then identifies the activity that corresponds with the greatest probability.

Next, the labelled data (training data) are processed and provide an output, which is a readout of the sum of all the inputs based on a pre-defined threshold. The context is then deduced from the labelled data.

In both cases, prediction consists of building a probabilistic model of the present observation given all past observations. Because the history of observations grows arbitrarily large it is necessary to limit the complexity of such a model and trim the results to the possible situations of interest.

Action execution

This module defines the rules that will trigger an action based on the context derivation. In order to do this, Bayesian probability rules are typically combined with first order predicate logic.[8]

Bayesian probability techniques are used to assign likelihood values for context identification. When in a specific context, a certain action must be triggered based on a certain probability and a threshold (Formula 1).

$$\forall\ \textit{derived context} \equiv \textit{if (associated probability of derived context} > \textit{threshold)} = \textit{TRUE} \Rightarrow \textit{execute associated action}$$

Formula 1. Basic action execution rule

A map between actions and context is defined and stored in the application. It should specify what is the threshold value that triggers the action and when this happens, i.e., right after the context is identified or after a certain time period

Some contexts might have several actions associated to them; in these cases, extra contextual information will be used in combination with the associated probability to choose the most appropriate action.

[8] Bayesian probability is can be seen as an extension of logic that enables reasoning with uncertain statements. It specifies some prior probability, which is then updated in light of new relevant data. First order logic is a the language describing the truth in mathematical formulas

3.2. The architecture design approaches

We analyse some of the architecture designs proposed by researchers and practitioners over the last decade[9] and summarise the main current approaches in the industry and their advantages and drawbacks.

In all the approaches examined, the emphasis is on designing an architecture that includes knowledge taxonomy and has the highest possible processing efficiency and programming simplicity. The main differences between the different design options are in where to place the context reasoning engine. From our survey, we have categorised practical context-aware architectures for mobile platforms into three types.

- Type 1 – layer based design: The context processing functionality is split between clearly defined layers that perform the three main tasks of understanding contextual information, processing it and using it in the application.

  In this approach, the integration of new context sources must be propagated through all the layers, increasing the complexity of maintenance and update.

- Type 2 – middleware design: The transformation of data into meaningful context information occurs in a single piece of middleware software. Contextual information is captured by sensors and directly analysed by the middleware reasoning engine. The separation between the middleware and the applications is unclear. There are two approaches for the middleware, it can be either centralised in a remote server or distributed over different devices of the mobile system.

  In this approach, the processing of context is slower because there is no pre-processing of data. The middleware is typically specific to a given context-aware situation.

- Type 3 – mixed design: A middleware and layer based framework that includes three components: a hardware abstraction layer that gets information from sensors, a context-manager that derives context information and maps it, and a privacy manager that selects information. The different layers communicate through a middleware context manager.

  This approach tries to bring more flexibility to the previous model by facilitating the task of integrating different context sources.

Overall, we find that there are some significant flaws in each of these architecture approaches. They are all ad-hoc designs that consider as given facts the locations of contextual sources, whether the sensors are local (part of the device) or remote, the amount of users of the system (one user or many) and the type of device where the application runs [14]. Therefore, all these designs are restricted in their scalability and performance. None of them state how to handle context changes over time or how new services can be implemented.

[9] Details on each of the architectures surveyed can be found at: 21. Korpipää, P. and J. Mäntyjärvi, *An Ontology for Mobile Device Sensor-Based Context Awareness.* Springer-Verlag, 2003: p. .451-458., 22. Henricksen, K. and J. Indulska. *A Software Engineering Framework for Context-Aware Pervasive Computing.* in *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications (PERCOM'04)* 2004., 23. Bardram, J.E., *Applications of ContextAware Computing in Hospital Work – Examples and Design Principles*, in *ACM Symposium on Applied Computing*. 2004., 24. Hinkley, K., et al., *Foreground and Background Interaction with Sensor-Enhanced Mobile Devices.* ACM Transactions on Computer-Human Interaction (TOCHI). , 2005. **Volume 12**(Issue 1 (March)): p. 31 - 52 , 25. Mika Raento, et al., *ContextPhone: A Prototyping Platform for Context-Aware Mobile Applications.* IEEE CS and IEEE ComSoc 2005, 2005., 26. Pauty, J., et al. *Research challenges in mobile and context-aware service development*. in *Proceedings of Future Research Challenges for Software and Services*. 2006., 27. Jie, S. and W. ZhaoHui, *Context Reasoning Technologies in Ubiquitous Computing Environment.* Book Series Lecture Notes in Computer Science Publisher Springer Berlin / Heidelberg ISSN 0302-9743 (Print) 2006. **4096/2006** , 14. Baldauf, M., S. Dustdar, and F. Rosenberg, *A survey on context-aware systems.* International Journal of Ad Hoc and Ubiquitous Computing, 2007. **2**(4): p. 263-277., 28. Eagle, N., A. Pentland, and D. Lazer, *Inferring friendship network structure by using mobile phone data.* PNAS. , 2009. **vol 106, no. 36**. and 18. Gómez-Barroso, J.L., et al., *Prospects of Mobile Search.* 2010, IPTS.

Moreover, the existing designs do not explain how they address the inherent restrictions of a mobile domain platform's nature: highly personal, always-on, anywhere and anytime access, need for real-time responses, processing of activities that are concurrent or have pauses between them, acknowledging the possible ambiguity of interpretation of context due to limitations of data capture, and, last but not least, the technical limitations of the platform itself (limited battery life, low processing power, potential unavailability or not affordable connectivity to the network). However, in the present early stage of development of most context-aware applications, these requirements have not yet been addressed properly or completely.

Therefore, in authors' view, new architecture designs for mobile context-aware systems should consider the following additional premises:

- a flexible architecture with interfaces to include new context sources easily, based on standard communication protocols and data definitions,
- a contextual information processing unit that performs a prior transformation of raw context into atoms that optimise the amount of historical data needed for reasoning and is as independent as possible of the platform where it runs (hardware and software), and of the type of context-aware situation,
- a light on-line machine learning algorithm that runs smoothly on a mobile platform (with limited processing power, memory capacity and energy constraints).

4. Context-awareness challenges

After analysing the underlying architecture and applications of context-aware systems, we review some of the inherent challenges. As explained earlier, context-aware systems architectures are made of a combination of hardware and software components, which can be either internal or external to the device where the application runs (Figure 5).

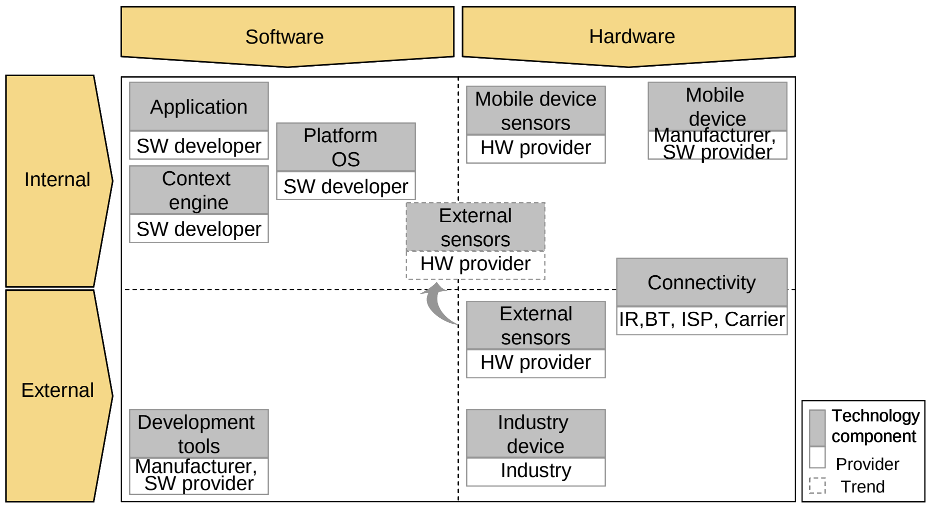

Figure 5. Context-aware ecosystem matrix

Each technology component is typically provided by a different manufacturer.

- Software applications that are context-aware enhanced are often provided by the developer.
- Context engines that recognise context and make use of it in the application, based on artificial intelligence techniques or reasoning are also provided by a (usually different) software developer.

- The mobile platform operating system (OS) where the application runs can either be an operating system or a web browser which are provided by software or computer enterprises.
- Sensors that capture contextual information can be inside or outside the mobile device where the application is running, although the trend is to embed as many of them as possible until the concept of Internet of things has been better developed and implemented. They are provided by diverse hardware manufacturers.
- The mobile device where the application runs (PDA, netbook, smartphone) is provided by a handset or computer manufacturer.
- Connectivity allows connection to other consumer electronic devices that can be located as close as 10 cm or thousands of kilometres away, by means of different types of connections, such as RFID, Bluetooth, infrared, WiFi, UMTS, LTE, WiMAX etc, which are provided by carriers, ISPs or sensor manufacturers (such is the case of RFID).
- Industry devices hold contextual information provided by consumer electronics manufacturers.
- Development tools are provided by a software development kit (SDK) to help developers, along with some additional tools to help in the programming of applications.

The obvious highly heterogeneous nature of context-aware applications components poses, from the authors' perspective, six main challenges to developers. These will be explained in the following sub-sections.

4.1 Sensor data interoperability and standards

The number of sensors for context-awareness increases every day. They use a myriad of proprietary schemes, standards and protocols. However, there are no widely adopted common solutions for data, formats and communication with them. In addition, manufacturers are launching mobile devices that have widely different implementations of sensing capabilities to capture potentially relevant contextual information.

Therefore, there is a considerable necessity for the industry to converge on standards and interoperability issues. These can come from an explicit agreement from the industry or from the market success of one or more platforms providing the level of integration required. In both cases, such convergence appears to remain far in the future.

4.2 Context data quality and heterogeneity

The problem with sensed contextual information is that it is highly dynamic (thus prone to noise and sensing errors), and while user-supplied information is reliable, on many occasions, it is out-of-date or of little use. This results in a lack of data quality which limits the performance of context-aware applications. The underlying issue is that two pieces of context information can take different values in terms of precision, correctness and trustiness [29]. Therefore, there is a need to perform a data completeness, verification and correction check [30]. This increases the complexity of application design and the computational burden.

In addition, the heterogeneity of contextual information also increases the complexity of the data analysis required when developing applications. Moreover, there is no unique solution for how to blend time and events to understand context changes. In summary, a logical model including all the existing components of a context-aware mobile computing environment does not exist. Therefore, the need arises at least to define a specific, context-aggregation mechanism [31] that includes data classification based on the nature of the context, and probably also to define a formal context-aware development language or interface development environment beyond the APIs available on a given platform's software development tool kits.

4.3 Early stage of reasoning techniques

Contextual reasoning techniques are at an early stage of development, combining cognitive and artificial intelligence to learn and predict context in environments that are mobile, have heterogeneous data, need on-line processing and are constantly changing. Current implementations are prone to wrong decision errors in which either actions do not take place in the right context (type 1 errors) or actions take place in the wrong context (type 2 errors). Therefore, this is still an active research area where new developments are expected.

4.4 Platforms technical limitations

Mobile devices now merge information technology, telecommunications and consumer electronics for new uses, which transform them into handheld computers. This increased mobile functionality demands more and more memory, fast performance and continuous wireless connectivity, in turn requiring higher energy consumption and processing power. This is one of the major challenges of any mobile user technology, not specific to context-awareness. However, this new type of applications will require high processing power and longer lived batteries to extract the relevant features of context, process it continuously, and maintain permanent connectivity with local wireless networks and with the mobile cloud.

4.5 Development frameworks

Currently, there are six main mobile platform OS, and many other less prominent ones, each with a different strategy and context-awareness approach (see table 3). There is no horizontal SDK framework compatible with all of them, in spite of some efforts to build cross-platform development tools. Obviously, the transaction costs imposed by this fragmentation are very high for interested developers.

Platform	Main constituents	Development model
Apple	iPhone-iPod-iPad + OS X App Store iTunes SDK	Closed model with tight control over hardware, software and applications
Nokia	Nokia devices Ovi Symbian / SDK	Increasingly open model with control of software and hardware development
Google	Nexus One + other devices Android marketplace Android / SDK	Open model with control of software development
RIM	Blackberry Blackberry Store RIM / SDK	Closed model with tight control over hardware, software and applications
Microsoft	Windows Marketplace Windows Mobile / SDK	Closed model with tight control over software development
Linux	Linux for mobile	Open model with loose control over software

Table 3: mobile platform OS strategy and context-awareness approach

There is a general trend to move towards a Linux core on a mobile OS base in open models with loose control over software development. An example of this is the "MeeGo" platform that supports flexibility when implementing new features in mobile devices, such as context-awareness [32].

4.6 Privacy issues

Privacy, along with security and data protection, is one of the major concerns for context-aware applications such as location based services [33]. In fact, one of the challenges in adapting applications to context is that even if the application is reliable, it can be considered by users as intruding [21]. Tools and methodologies are needed to define a design of suitable privacy policies for application development in order to protect users from possible abuses. Several strategies are under discussion, such as privacy by design (i.e., requiring intervention and explicit acceptance from the user), privacy by law (i.e., defining the sphere of personal data not subject to use by applications) or user empowerment (i.e., keeping users in complete control of their personal data, able to switch on and off the application and data). One way to avoid privacy issues is to define an architecture in which the context capturing and processing is done on the mobile device rather than on external platforms. This ensures that the user's data are not shared with external parties.

5. Conclusions

Context-aware computing enables the targeting and personalising of applications through contextual information. However, making context-aware applications a reality entails considerable challenges for the current schemes of development and implementation of mobile applications. It is necessary to confront the heterogeneous nature of contextual information scattered in many distinct and much different sources, the fragmentation of platforms, and a number of technological elements still under development. In addition, data acquisition, modelling and processing are rather different from regular mobile applications because they blend physical with digital data in a ubiquitous environment that requires on-line personalised responses, should be respectful of privacy concerns and should keep the users in control. In addition, mobile platforms have technical limitations for context recognition tasks that may be hugely demanding in memory and processing power, as well as for integrating themselves with sensing devices that might have non-standard connectivity methods and may not be fully deployed.

All these factors imply the need for a new architecture and tools oriented to capture context and process it, suited to the particularities of the domain and able to evolve with it. In this paper, the main examples of mobile context-aware applications have been analysed, exploring the architectures proposed to support them as well as the underlying technologies. The paper has outlined the main challenges that contextual application development poses to the industry.

In summary, from the authors' perspective, further analysis is needed to gain in-depth understanding of mobile context-awareness opportunities. On the technical side, it is necessary to establish some standards for data acquisition and interoperability, solve the challenges in data quality and heterogeneity, by-pass the technical limitations of given platforms, improve reasoning techniques, increase the stability and reduce the number of possible development frameworks, and address privacy issues, properly. However, not only the technical side is important to the prospects of mobile context-awareness. The ecosystem is still notoriously immature, with many niches empty (new sensors, context-aggregators, context platforms, providers of new tools for context, etc.), as suggested by the brief examination presented in this paper. Also many of the main techno-economic aspects are still largely unknown: the value proposition to users, users' true demands and expectations, profitable and scalable business models, strategies of the main players interested in the domain, etc. However, all of these uncertainties should not discourage innovators and entrepreneurs; on the contrary, for the next years this area will be the wild frontier of mobile-ICT development, where high rewards await those who dare to confront it, and, ultimately, society at large.

References

1. Sørensen, C.-F., et al., *a Context-Aware Middleware for Applications in Mobile Ad Hoc Environments*, in *Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing*. 2004, ACM International Conference Proceeding Series: Toronto, Ontario, Canada p. 107 - 110.

2. Clark, W., *Key issues for contex/aware computing 2010*, in *Research*, Gartner, Editor. 2010.

3. Dargie, W., *Context and self-management*, in *Context-aware computing and self-management systems*, W. Dargie, Editor. 2009, Chapman & hall book Boca Raton, FL. p. 5-11.

4. Bouquet, P., L. Serafini, and R. H.Thomason, *Perspectives on contexts*. 2008: CSL Publications. 285.

5. Corp., i. *Shipments of Cell Phone Motion Sensors to Rise Fivefold by 2014*. 2010 5 May 2010 [cited 2010 22 May 2010]; Available from: http://www.cellular-news.com/story/43168.php.

6. Ramos, S., C. Feijóo, and J. Gomez-Barroso, *Next Generation Mobile Network Deployment Strategies.* Journal of the Institute of Telecommunications Professionals, 2009. **3**(1): p. 13-19.

7. Grauballe, A., G.P. Perrucci, and F.H.P. Fitzek, *Introducing Contextual Information to Mobile Phones by External and Embedded Sensors*, in *International workshop on mobile device and urban sensing MODUS08*. 2008: St. Louis,MO

8. Juniper, *Mobile augmented reality. A whole new world*. 2009

9. Crowley, J.C.a.J.L., *Context is key.* Communication of ACM, 2005.

10. Coutaz, J. and J.L. Crowley, *Context is key.* Communication of ACM, 2005. **48**(3): p. 49 - 53.

11. Gellersen, H.W., A. Schmidt, and M. Beigl, *Multi-Sensor Context-Awareness in Mobile Devices and Smart Artifacts.* Mobile Networks and Applications, 2002. **7**(5): p. 341–351.

12. Lassila, O. and D. Khushraj. *Contextualizing Applications via Semantic Middleware*. in *Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05) 2005 IEEE*. 2005.

13. Chen, G. and D. Kotz, *Solar: A pervasive-computing infrastructure for context-aware mobile applications*, in *Dartmouth Computer Science Technical Report TR2002-421*. 2002, Dartmouth College. .

14. Baldauf, M., S. Dustdar, and F. Rosenberg, *A survey on context-aware systems.* International Journal of Ad Hoc and Ubiquitous Computing, 2007. **2**(4): p. 263-277.

15. Barkhuus, L. and A. Dey, *"Is context-aware computing taking control away from the user? Three levels of interactivity examined".* Proceedings of Ubicomp 2003.

16. Anumba, C. and Z. Aziz, *Case Studies of Intelligent Context-Aware Services Delivery in AEC/FM.* Springer, 2006. **4200/2006**(LNCS: Intelligent Computing in Engineering and Architecture): p. 23-31.

17. De Souza e Silva, A., *Alien revolt (2005-2007): a case study of the first location-based mobile game in Brazil.* IEEE Technology and Society Magazine, 2008. **27((**Spring 2008): p. 18-28.

18. Gómez-Barroso, J.L., et al., *Prospects of Mobile Search*. 2010, IPTS.

19. Sharon, M.E. Facebook Blog 2010 [cited august 24 2010]; Available from: http://blog.facebook.com/blog.php?post=418175202130.

20. Schmidt, A. and K.V. Laerhoven, *"How to Build Smart Appliances?".* IEEE Personal Communications, 2001.

21. Korpipää, P. and J. Mäntyjärvi, *An Ontology for Mobile Device Sensor-Based Context Awareness.* Springer-Verlag, 2003: p. .451-458.

22. Henricksen, K. and J. Indulska. *A Software Engineering Framework for Context-Aware Pervasive Computing*. in *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications (PERCOM'04)*
2004.

23. Bardram, J.E., *Applications of ContextAware Computing in Hospital Work – Examples and Design Principles*, in *ACM Symposium on Applied Computing*. 2004.

24. Hinkley, K., et al., *Foreground and Background Interaction with Sensor-Enhanced Mobile Devices.* ACM Transactions on Computer-Human Interaction (TOCHI). , 2005. **Volume 12**(Issue 1 (March)): p. 31 - 52

25. Mika Raento, et al., *ContextPhone: A Prototyping Platform for Context-Aware Mobile Applications.* IEEE CS and IEEE ComSoc 2005, 2005.

26. Pauty, J., et al. *Research challenges in mobile and context-aware service development*. in *Proceedings of Future Research Challenges for Software and Services*. 2006.

27. Jie, S. and W. ZhaoHui, *Context Reasoning Technologies in Ubiquitous Computing Environment.* Book Series Lecture Notes in Computer Science Publisher Springer Berlin / Heidelberg ISSN 0302-9743 (Print) 2006. **4096/2006**

28. Eagle, N., A. Pentland, and D. Lazer, *Inferring friendship network structure by using mobile phone data.* PNAS. , 2009. **vol 106, no. 36**.

29. Buchholz, T., A. Kupper, and M. Schiffers. *Quality of context: What it is and why we need it.* . in *Proceedings of the Workshop of HP OpenView University Association*. 2003. Geneva: HP.

30. Bernardos, A.M., P. Tarrío, and J.R. Casar, *A data fusion framework for context-aware mobile services.* , in *Proceedings of the IEEE International Conf. in Multisensor Fusion and Integration for Intelligent Systems*. 2008, IEEE Computer Society. p. 606-613,.

31. Lapkin, A. (2009) *Gartner Says Context-Aware Computing Will Provide Significant Competitive Advantage*. Press room **Volume**,

32. LinuxFoundation, T. *MeeGo*. 2010 [cited; Available from: http://meego.com/about.

33. Feijoo, C., et al., *The Next Paradigm Shift in the Mobile Ecosystem: Mobile Social Computing and the Increasing Relevance of Users.* COMMUNICATIONS & STRATEGIES, 2009. **75**(3rd quarter): p. 57-77.