Gronli, T, Ghinea, G and Younas, M

Context-aware and automatic configuration of mobile devices in cloud-enabled ubiquitous computing.

Gronli, T, Ghinea, G and Younas, M (2014) Context-aware and automatic configuration of mobile devices in cloud-enabled ubiquitous computing. *Personal and Ubiquitous Computing*, 18 (4). pp. 883-894.

doi: 10.1007/s00779-013-0698-3

Context-aware and Automatic Configuration of Mobile Devices in Cloud-enabled Ubiquitous Computing

Tor-Morten Grønli
Norwegian School of IT
Oslo, Norway
tmg@nith.no

Gheorghita Ghinea
Brunel University
London, UK
george.ghinea@brunel.ac.uk

Muhammad Younas
Oxford Brooks University
Oxford, UK
m.younas@brookes.ac.uk

Abstract

The paper explores the notion of cross-source integration of cloud-based, context-aware information in ubiquitous computing. Moreover, the described solution incorporates remote and automatic configuration of Android phone and advances the research area of context-aware information. The purpose of this work is to innovate and expand the notion of context-awareness in order to enable automatic adaptation and behaviour altering in accordance with implicit user needs. The novelty of this work is the new proposed solution and design whereby context-aware information is harvested from several dimensions to build rich foundation on which algorithms for context-aware computation can be based. Design research was applied as the methodology for pursuing this research. Through real life evaluation, application artifacts were tested, evaluated and verified. The experiment was conducted with testing on a group of real world users. Results from the conducted experiment show the viability of tailoring contextual information to provide user with timely, relevant and adapted application behavior and content. Through a developed application suite, we create a remote configurable, adaptive Android phone application to demonstrate our proposal. Through discussion and user evaluation we show the feasibility of such an approach and elaborate on how this can take the tailored user experience to the next level.

INTRODUCTION

The notion of context-aware computing is generally the ability for the devices to adapt their behaviour to the surrounding environment, and ultimately enhancing usability (Dey and Abowd, 1999). In particular, context-awareness is becoming an important factor when it comes to mobile devices, as users bring their mobile phone or tablet just about everywhere, highlighting the need for adapting the content to the users current situation. Dey and Abowd (1999) have remarked that if we fully understand context in a given environment and setting we are better able to adapt the context-aware behaviour in our applications. The consequence of this is devices that adapt content based on the user's context, eliminating the need for users to manually do these tasks. Indeed, context is a major part of our daily life and support for sharing and using contextual information will improve user interaction. As such, it is important to take advantage of this by viewing people as consumers of the information and not other systems.

Context-aware solutions have been around for some years, and have been used in a variety of applications (Baldauf et al 2007), (Kapitsaki et al 2009), (Younas and Awan, 2013). In this work we seek to build further on these achievements and utilize context as a source of information for user information and interface tailoring. The issue with much of the earlier approaches is that they have only looked at one source of context-aware information or more than one source, but utilized separately. We propose a different approach, where we combine context-aware information from several dimensions to build a rich foundation and to base our algorithms on. Further, we exploit cloud computing technology to create a new user experience and a new way to invoke control over user's mobile devices. We combine context information received both from the phone itself and information retrieved from cloud-based servers. Our solution is a remote configuration of an Android phone, which uses the context-aware information foundation to constantly adapt further on and change in accordance with the user's implicit requirements. All data is integrated to create a context-aware mobile device, where we implemented a new customized home screen application for Android enabled devices.

The paper is structured as follows. The next section explains the basic concepts and the underlying technologies of the proposed approach. Related work is then described before the design and development of the proposed approach is presented. Furthermore, results and evaluation are presented before the paper is concluded.

TECHNOLOGICAL ASPECTS

Context-aware computing builds on a combination of various technologies such as computers, mobile devices, human, sensors, cloud, web, Internet and software services. All such technologies and services are interconnected in order to communicate with each other and exchange of useful information such as location, weather information, traffic conditions, road direction, and health and safety. In the following we briefly describe the main technologies that have been used in the design and development of the proposed approach:

Mobile Devices

This research was primarily tested on two mobile devices, the HTC Nexus One and the HTC Evo.

The HTC manufactured Nexus One represents one of the first world wide available commercial Android phones. The Nexus One features dynamic voice suppression and has a 3.7-inch AMOLED touch sensitive display supporting 16M colours with a WVGA screen resolution of 800 x 480 pixels. It runs the Google Android operating system on a Qualcomm 1 GHz Snapdragon processor and features 512 MB standard memory, 512 MB internal flash ROM and 4 GB internal storage.

The HTC Evo 4G is an Android phone shipped by the Sprint operator for the American CDMA network. The HTC Evo 4G features a 4.3- inch TFT capacitive touchscreen display supporting 64K colours with a screen resolution of 480 x 800 pixels. It runs the Google Android operating system on a Qualcomm 1 GHz Scorpion processor and features WI-FI 802.11b/g, 512 MB standard memory, 1 GB internal flash ROM and 8 GB internal storage.

Sensors

Sensors are an important source of information input in any real world context and several previous research contributions look into this topic. For instance, Parviainen et al. (2006) approached this area from a meeting room scenario. They found several uses for a sound localization system, such as: automatic translation to another language, retrieval of specific topics, and summarization of meetings in a human-readable form. In their work, they find sensors a viable source of information, but also acknowledge there is still work to do, like improving integration. Modern smart phones have are a number of built in sensors. For example, in our test phone (the HTC Nexus One) we had 5 sensors available: accelerometer, magnetic field, orientation, proximity and light. All of these sensors can be accessed as local services through well-defined APIs in the Android platform. In our work, by taking the aspect of sensors and context-awareness and integrating it in a mobile application we reduce the workload for the end users. Thus, in combination with a cloud-based server application we are able to remotely configure the mobile devices independent of the device types. This creates a new concept of context-awareness and embraces the user in ways previously unavailable.

Cloud Computing

Cloud computing has received considerable attention in the software industry. It is a buzzword frequently used for all sorts of services, ranging from hosted virtual machines to simple web based email applications. So what is cloud computing? We have used the definition created by NIST (Mell and Grance, 2011) (National Institute of Standards and Technology, United States): "*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources …*". Large IT companies like Microsoft, Google and IBM, all have initiatives relating to cloud computing (Parviainen et al., 2006) which have spawned a number of emerging research themes, among which we mention: *cloud system design* (Mell and Grance, 2011), *benchmarking of the cloud* (Mei et al., 2008) and *provider response time comparisons*. Mei et al. (2008) have pointed out 4 main research areas in cloud computing that they find particularly interesting is the *Pluggable computing entities, data access transparency, adaptive behaviour of cloud applications and automatic discovery of application quality.*

Our work focuses on *data access transparency*, where clients transparently will push and pull for data from the cloud, and *adaptive behaviour of cloud applications*. We adapted the behaviour of the Google App Engine server application based on context information sent from the users' devices thus integrating context and cloud on a secure (Binnig et al., 2009) mobile platform.

Google App Engine

The Google App Engine makes it possible to run web applications on the Google infrastructure. App Engine applications are easy to build, maintain and scale (Google, 2012). The Google App Engine is a cloud-based platform as a service (PaaS); the platform is pre-configured by Google and provides a much higher abstraction than an Infrastructure as a Service (IaaS) platform such as the Amazon elastic cloud. The App Engine is well integrated with Google Apps services like docs, Gmail and authentication. The platform imposes certain limitations on the developers, for example no threading API, but by enforcing these Google is able to provide very high scalability. The Google App Engine supports runtime environments for Java, Python and Go. Each environment provides standard protocols and common technologies for web application development (Google, 2012). The Google App Engine has been used as the main server installation for this research and has proven to be a stable environment for deploying the Java server applications without having to consider hardware requirements and software installations.

THE PROPOSED APPROACH

The proposed approach utilizes web resources for effectively tailoring the user experience in cloud-based and context-aware mobile settings. Earlier approaches have either looked at one source of context-aware information or in case of multiple contexts, the information is used separately (give some references related to the earlier work). We propose a different approach, where we combine context-aware information from several dimensions to build a rich foundation to base our algorithms on. This will allow for cloud computing to be used to create new user experiences and new ways to invoke control over a smartphone. With this a basis, it is possible to have an always adapting user interface.

To have a feature rich, scalable and service oriented server framework, a cloud computing solution was chosen. Traditional REST framework services were considered, but they were found to be insufficient scalable and extensible to add and remove context-aware sources in an ad hoc manner. The cloud-based approach has also the advantage of being run as a platform-as-a-service instance in the separate hosting instance of Google App Engine.

Design and Implementation

We have implemented an application suite, which provides a fully functional demonstration of the system. One of the main technical goals with the system was to make the interaction between the cloud and the mobile device as seamless as possible for the user.

The system was designed with three major components: an android client, a cloud server application, and the remote Google services. Figure 1 gives an overview of the implementation of the system (blue boxes in the diagram represent the parts of the system we created). The white boxes, like Google calendar and contacts, are external systems we communicated with. The server application was deployed remotely in the cloud on the Google App Engine, whilst data was also stored remotely in Google cloud services.

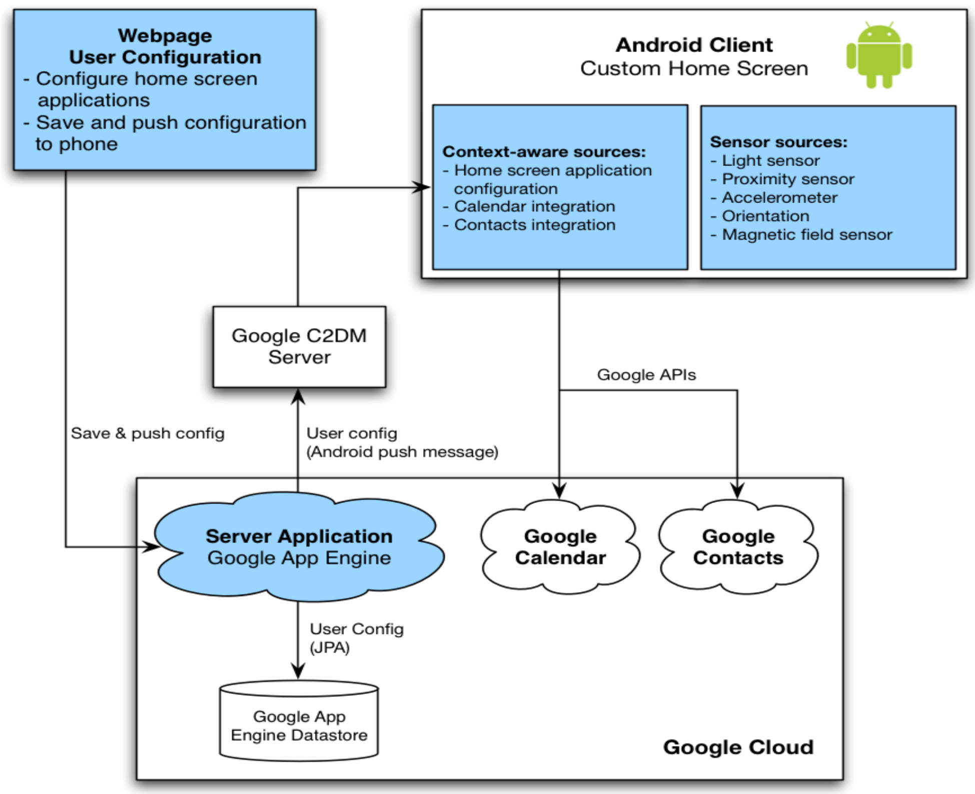


Figure 1: Application suite architecture

After the Android client was installed on the mobile device, the device will register itself to the Google services as illustrated in the code snippet.

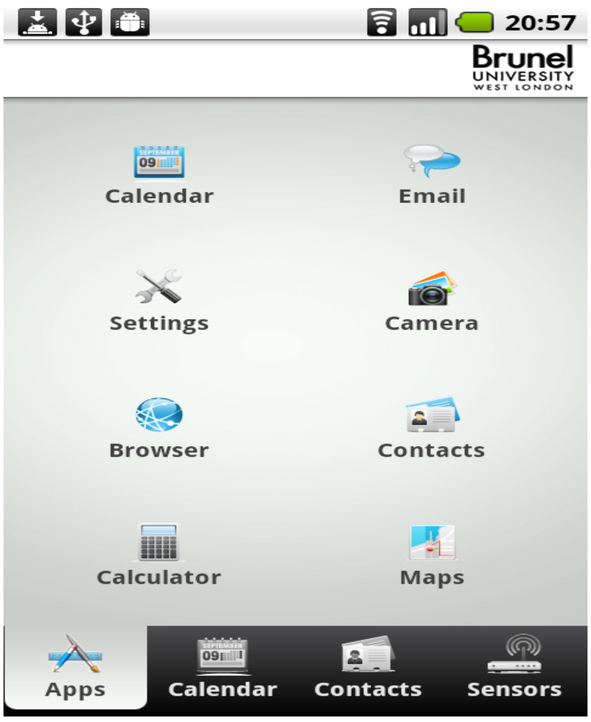
The users would start by logging in to the webpage. This webpage is part of the server application hosted on the Google App Engine. The login process uses the Google username/password. By leveraging the possibilities with Open Authorization (OAuth) we facilitated for the user sharing of their private calendar appointments and contacts stored in their Google cloud account without having to locally store their credentials. OAuth allowed us to use tokens as means of authentication and made it thereby possible for us to act as a third party granted access by the user.

After a successful authentication the user is presented with a webpage showing all configuration options. Because the configuration for each user is stored in the cloud, we avoided tying it directly to a mobile device. One of the major benefits of this feature is that the user did not need to manually update each device; they have a “master configuration” stored externally that can be directly pushed to their phone or tablet. It is also easier to add more advanced configuration options when the user can take advantage of the bigger screen, mouse and keyboard on a desktop/laptop PC for entering configuration values than those found on mobile devices. On the webpage, by selecting the applications s/he wants to store on the mobile device and pressing the “save configuration”-button, a push message is sent to the client application.

Android client

The client application was implemented on an Android device. This application utilized context-aware information from the device in the form of time, location and sensors. Additionally it utilized context-aware information from the cloud-integrated backend to acquire dynamic interface content, contacts and calendar data. At launch, the application would look as illustrated in Figure 2.

Figure 2: Home screen interface at launch of application

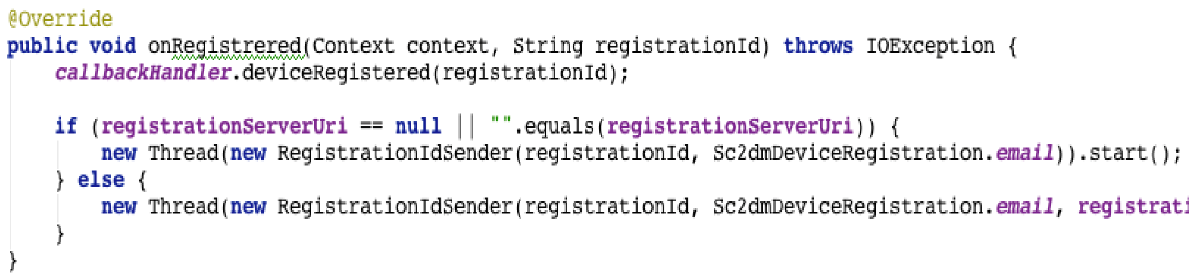


This interface would change depending on the user's given context. The applications available would be adapted and customized to match the current computed user context and thereby, unobtrusively alter the user experience.

Cloud to Device Messaging

The message is sent with a push feature for Android called C2DM (Cloud to Device Messaging), available from Android 2.2. The C2DM feature requires the Android clients to query a registration server to get an ID that represents the device. This id is then sent to our server application and stored in the Google App Engine datastore. When a message needs to be sent, the "save configuration"-button is pushed. We composed the message according to the C2DM format and sent it with the registration id as the recipient. These messages are then received by the Google C2DM servers and finally transferred to the correct mobile device. A snippet from this process is shown below (figure 3).

Figure 3: Excerpt from the device messaging process

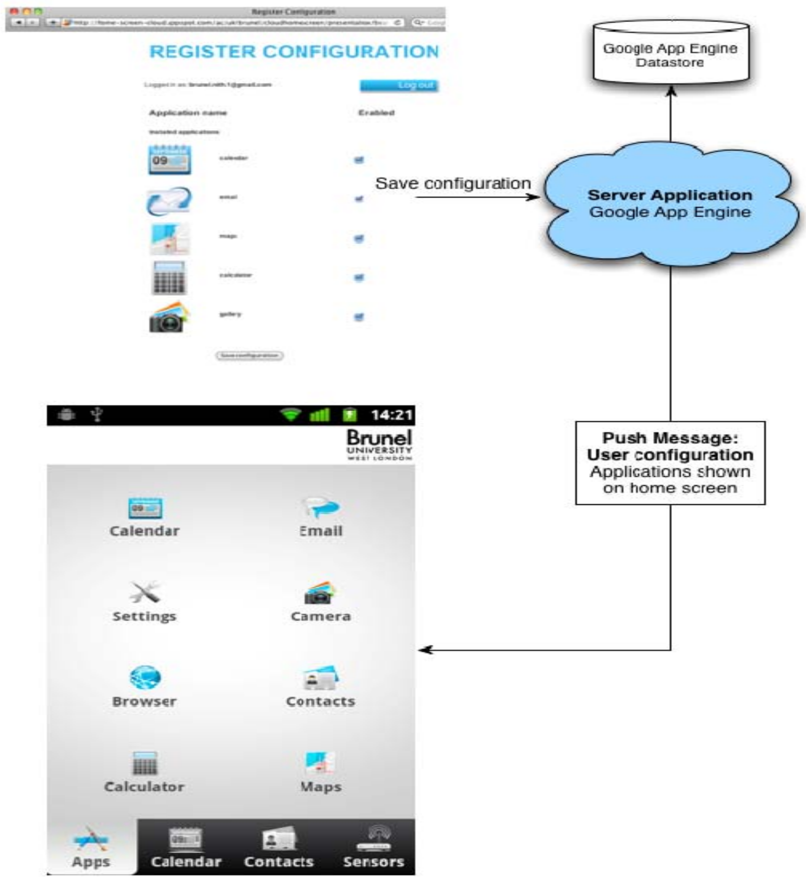


```
@Override
public void onRegistrered(Context context, String registrationId) throws IOException {
    callbackHandler.deviceRegistered(registrationId);

    if (registrationServerUri == null || "".equals(registrationServerUri)) {
        new Thread(new RegistrationIdSender(registrationId, Sc2dmDeviceRegistration.email)).start();
    } else {
        new Thread(new RegistrationIdSender(registrationId, Sc2dmDeviceRegistration.email, registrati
    }
}
```

The C2DM process is visualized in the figure below (figure 4). This technology has a few very appealing benefits: messages can be received by the device even if the application is not running, saves battery life by avoiding a custom polling mechanism, and takes advantage of the Google authentication process to provide security.

Figure 4: C2DM message cycle


REGISTER CONFIGURATION
Save configuration
Google App Engine Datastore
Server Application Google App Engine
Push Message: User configuration Applications shown on home screen
Brunel UNIVERSITY WEST LONDON
14:21
Calendar
Email
Settings
Camera
Browser
Contacts
Calculator
Maps
Apps
Calendar
Contacts
Sensors

Our experience with C2DM was mixed. It is a great feature when you get it to work, but the API is not very developer friendly. This will most likely change in the future since the product is currently in an experimental state, but it requires the developer to work with details like device registration and registration id synchronization. Although C2DM does not provide any guarantees when it comes to the delivery or order of messages, we found the performance to be quite good in most of the cases. It is worth mentioning that we did see some very high spikes in response time for a few requests, but in the majority of cases the clients received the responses within about half a second. Performance measurements we recorded, while doing the user experiments, were on average 663 milliseconds of response value. It is also important to note that issues like network latency will affect the performance results.

The calendar and contacts integration was also an important part of the Android application. We decided to allow the Android client to directly send requests to the Google APIs instead of going the route through the server. The main reason for this is that we did not think the additional cost of the extra network call was justified in this case. The interaction is so simple and there is very little business logic involved in this part so we gave the clients the responsibility for handling it directly. The implementation worked by simply querying the calendar and contact API and then using xml parsers to extract the content.

Meta-tagging

To make it possible for users to tag their appointments and contacts with context information we added special meta-tags. By adding a type tag, for example *$[type = work]* or *$[type= leisure]*, we were able to know if the user had a business meeting or a leisure activity. We then filtered the contacts based on this information. If the tag *$[type=work]* was added, this lets the application know that the user is in a work setting and it will automatically adapt the contacts based on this input. In a work context only work related contacts would be shown. To add and edit these tags we used the web-interface of Google contacts and calendar.

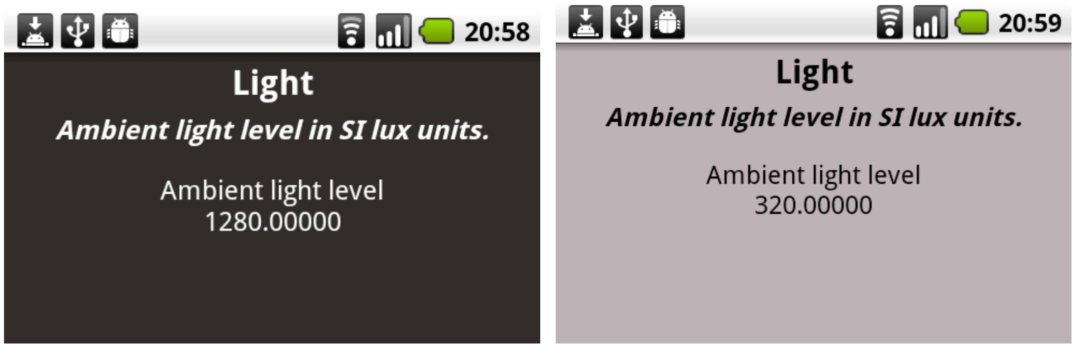
Sensors as input data

The sensors on the mobile device were also used as input to the application. We used the API available on the Android platform and through a base class called *SensorManager* we were able to access all of the built-in sensors on the mobile device. Working with this API was a very pleasant experience. It does what you except it to and is simple to use.

We started out by just showing the input values from the sensors in our pilot study. When we expanded the application after the initial tests we wanted to use the sensor input to further enhance the user experience. We ended up with using two features directly in the application, the accelerometer and the light sensor. The accelerometer was used to register if the device was shaking. If the device is shaking it means that the user is probably on the move, for example running or walking fast. In these cases, we automatically change the user interface to a much simpler view that has bigger buttons and is easier to use when on the move.

The second sensor we used in our experiment was the light sensor. By constantly registering the lighting levels in the room we adjusted the background colour of the application (figure 5). We changed the background colour of the application very carefully, as it would be very annoying for the users if colour changes were happening often and were drastic. Accordingly, we gradually faded colour when the lighting values measured from the environment changed.

Figure 5: Background light adjustment



20:58

Light

Ambient light level in SI lux units.

Ambient light level
1280.00000

20:59

Light

Ambient light level in SI lux units.

Ambient light level
320.00000

RESULTS

The research experiment had two main phases of evaluation. In phase one, a pilot test was performed with a total of 12 users. These users were of mixed age, gender and computer expertise. The results from this phase were fed back into the development loop, as well as helped remove some unclear questions in the questionnaire. In the second phase, the main evaluation, another 40 people participated. Out of the 40 participants in the main evaluation, two did not complete the questionnaire afterwards and were therefore removed making the total number of participants 38 in the main evaluation. All 12 from the pilot test and the 38 from the main test session were aged between 20 and 55 years old. All participants had previous knowledge of mobile phones and mobile communication, but had not previously used the type of application employed in our experiment. None of the pilot test users participated in the main evaluation. From the user computer experience classification (asserted based on a questionnaire employing the taxonomy of McMurtrey (2001)) we learnt that the majority of the users had a good level of computer expertise.

The results presented here illustrate different parts of the questionnaire: statements one to three target the *user interface*, statements four to six regard *sensor integration*, statements seven to nine focus on *the web application*, statements ten to thirteen centre on *context-awareness,* while statements fourteen to seventeen are about *cloud computing.* The questionnaire ends with *overall usefulness,* which is in an open-ended question for comments. The statements are given below (Table 1) together with the mean, standard deviation and the results of applying a one-sample t-test.

Table 1: User evaluation questionnaire and results

	Statement	Mean	Std. Dev.	t-test
Statements regarding user interface				
Statement 1	It is easy to see the available functions	3.50	0.51	.000
Statement 2	The features of the application are hard to use	1.89	0.73	.378
Statement 3	The adaptability of the application is a feature WE approve	3.45	0.55	.000
Statements regarding sensor integration				
Statement 4	The background colour in the application changes when the lighting in the room changes	3.55	0.65	.000
Statement 5	When moving around, a simplified user interface is not presented	2.11	1.06	.544
Statement 6	WE find sensor integration annoying and would disable it on my device	1.84	0.72	.183
Statements regarding the web application				
Statement 7	WE was able to register my device application configuration in the web application	3.61	0.59	.000
Statement 8	WE was not able to store and push my configuration to my mobile device from the web page	1.47	0.80	.000

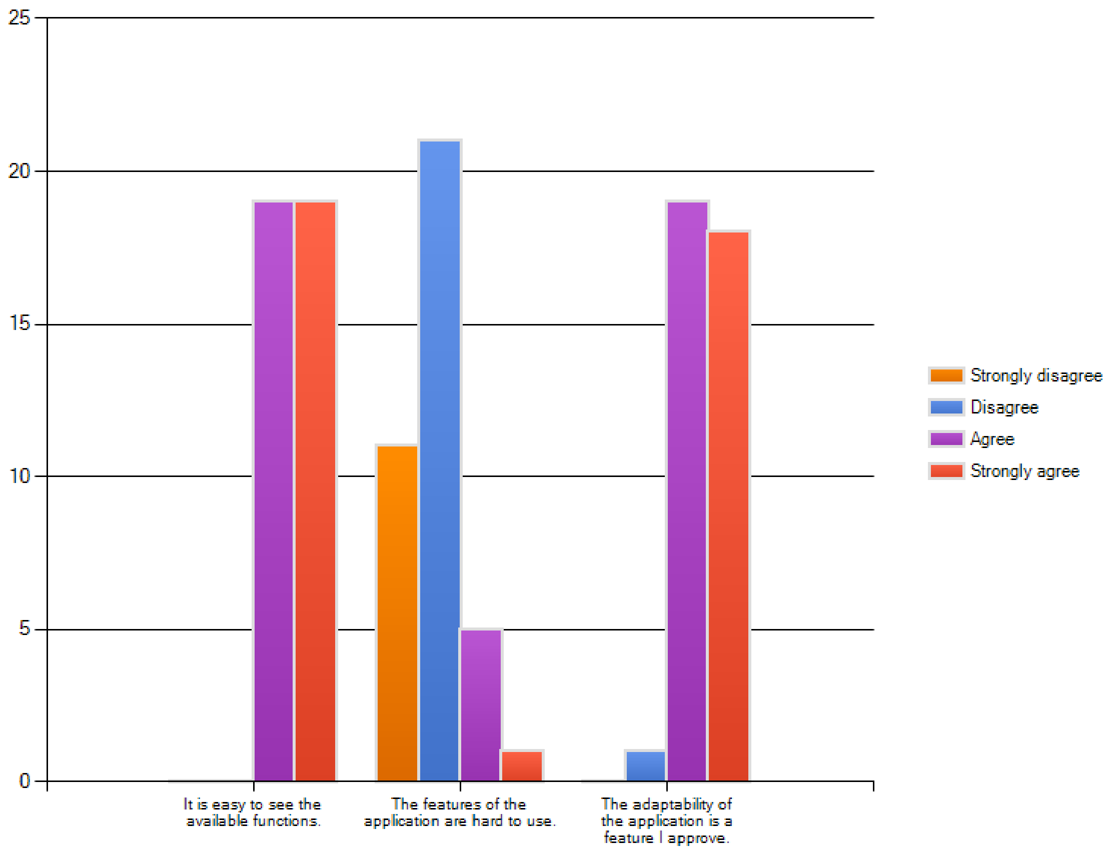
Statement 9	WE would like to configure my phone from a cloud service on a daily basis, (webpage user config and Google services like mail/calendar/contacts)	3.18	0.69	.000
Statements regarding context-awareness				
Statement 10	The close integration with Google services is an inconvenience, WE am not able to use the system without changing my existing or creating a new e-mail account at Google	1.76	0.88	.107
Statement 11	Calendar appointments displayed matched my current user context	3.58	0.55	.000
Statement 12	The contacts displayed did not match my current user context	1.29	0.52	.000
Statement 13	WE would like to see integration with other online services such as online editing tools (for example Google Docs) and user messaging applications (like Twitter and Google Buzz)	3.29	0.73	.000
Statements regarding cloud computing				
Statement 14	WE do not mind Cloud server downtime	2.08	0.78	.539
Statement 15	WE do not like sharing my personal information (like my name and e-mail address) to a service that stores the information in the	2.16	0.79	.225

	cloud			
Statement 16	Storing data in the Google Cloud and combining this with personal information on the device is a useful feature.	3.26	0.60	.000
Statement 17	WE find the cloud-to device application useful	3.53	0.51	.000
Open comment question				

User interface

Statements one to three deal with the user interface (Figure 6). These results reveal positive facts about the interface, highlighting that the majority of the users found it easy to see all available functions; moreover the vast majority (37/38) also finds the adaptability of the application a feature that they approve of. However, looking at statement two, their opinions are split in terms of whether the features are hard to use, and the statement results are statistically not significant.

Figure 6: Statements regarding user interface

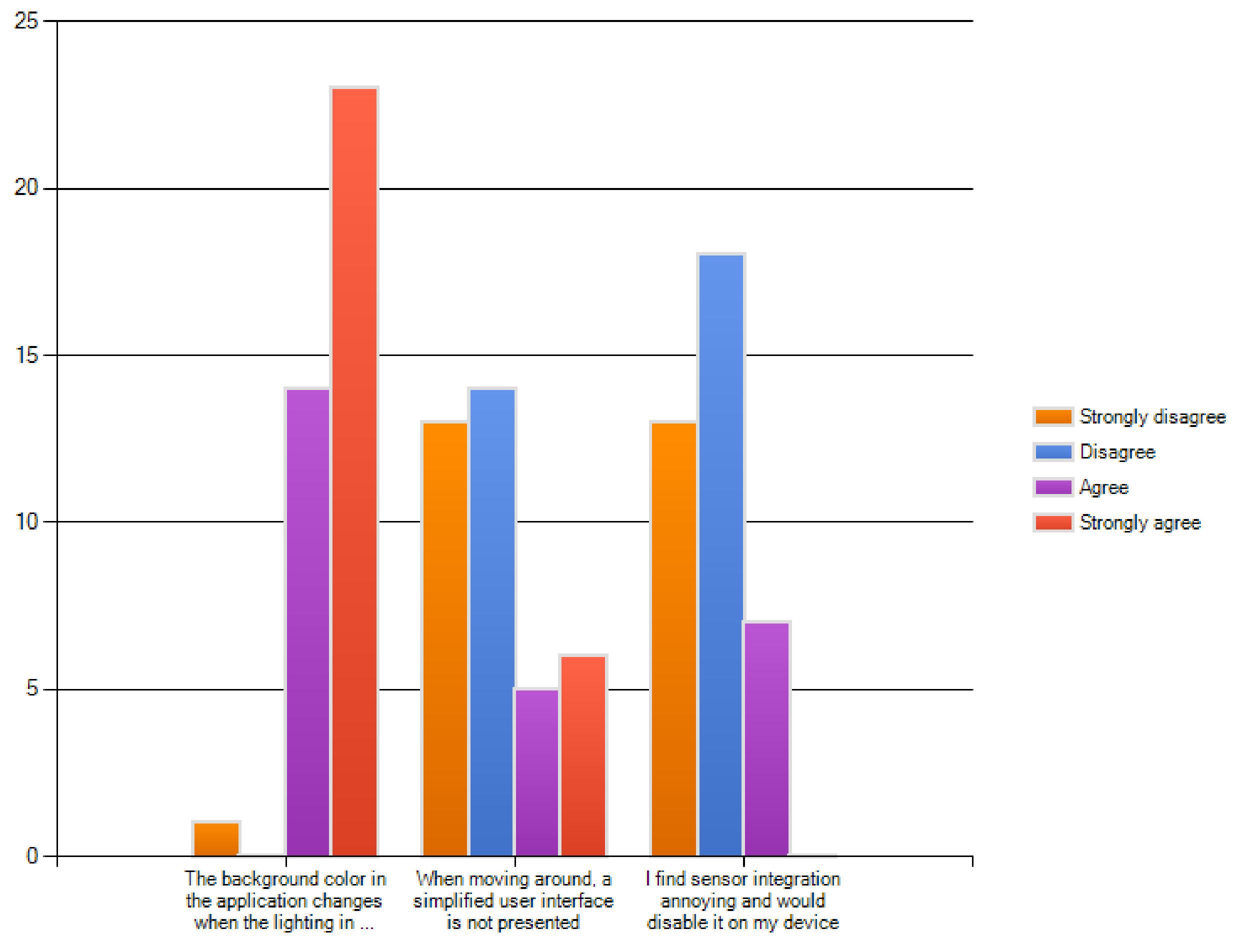


The results indicate that it is easy to get an overview of the application and the test candidates find adaptability a positive feature.

Sensor integration

Opinions are split regarding sensor integration. Users agree that the light sensor is working as expected, but disagree whether the simpler user interface changes. This can be due to the sensitivity threshold programmed for the sensor, and should be verified by more comprehensive testing. The majority, 32 out of 38, would not deactivate sensor integration and this is a useful observation, highlighting that sensors should be further pursued as context-aware input.

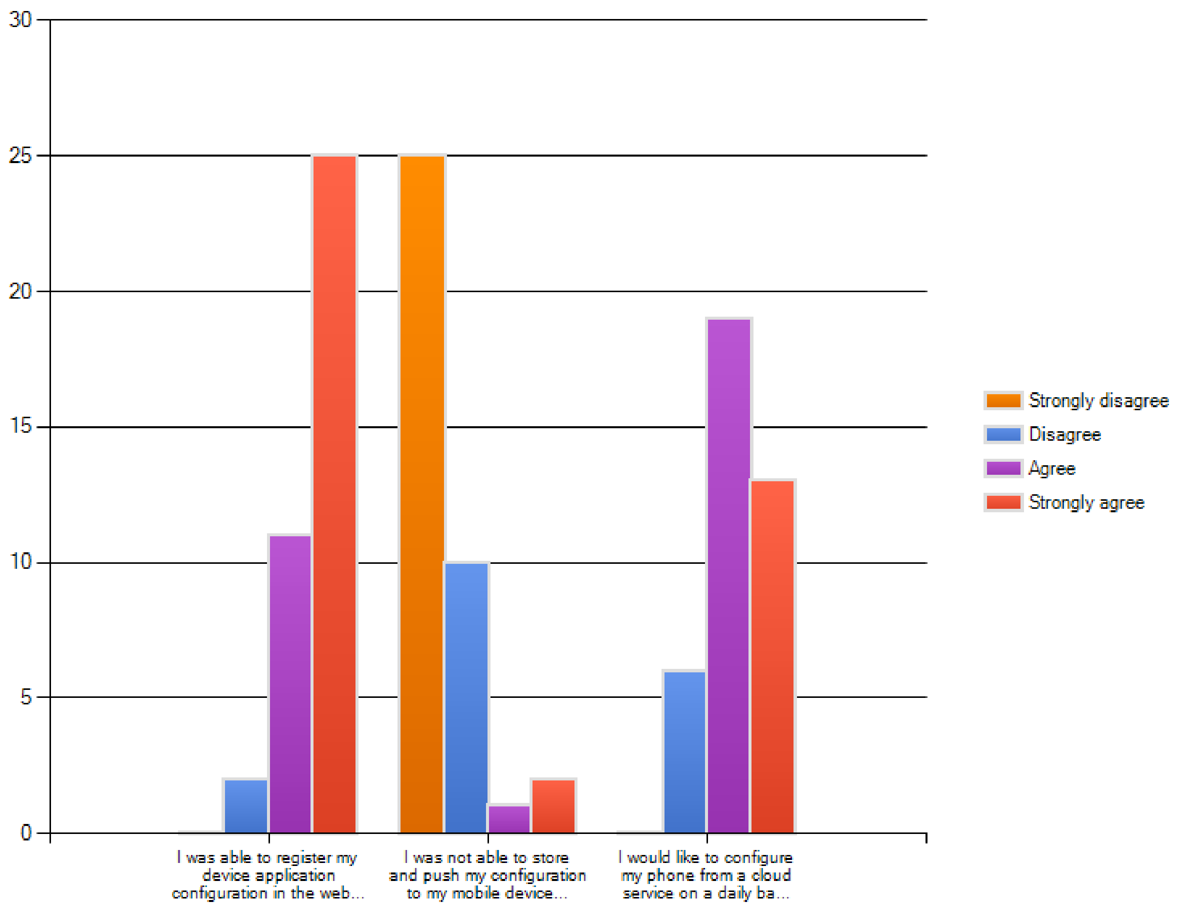
Figure 7: Statements regarding sensor integration



Web application

The statements dealing with the web application at Google App Engine (Figure 8) show that the web application performed as expected, by letting participants register their devices as well as pushing performed configurations to the devices. Also answers from statement nine are quite interesting, highlighting a positive attitude towards cloud-based services (32 out of 38 are positive).

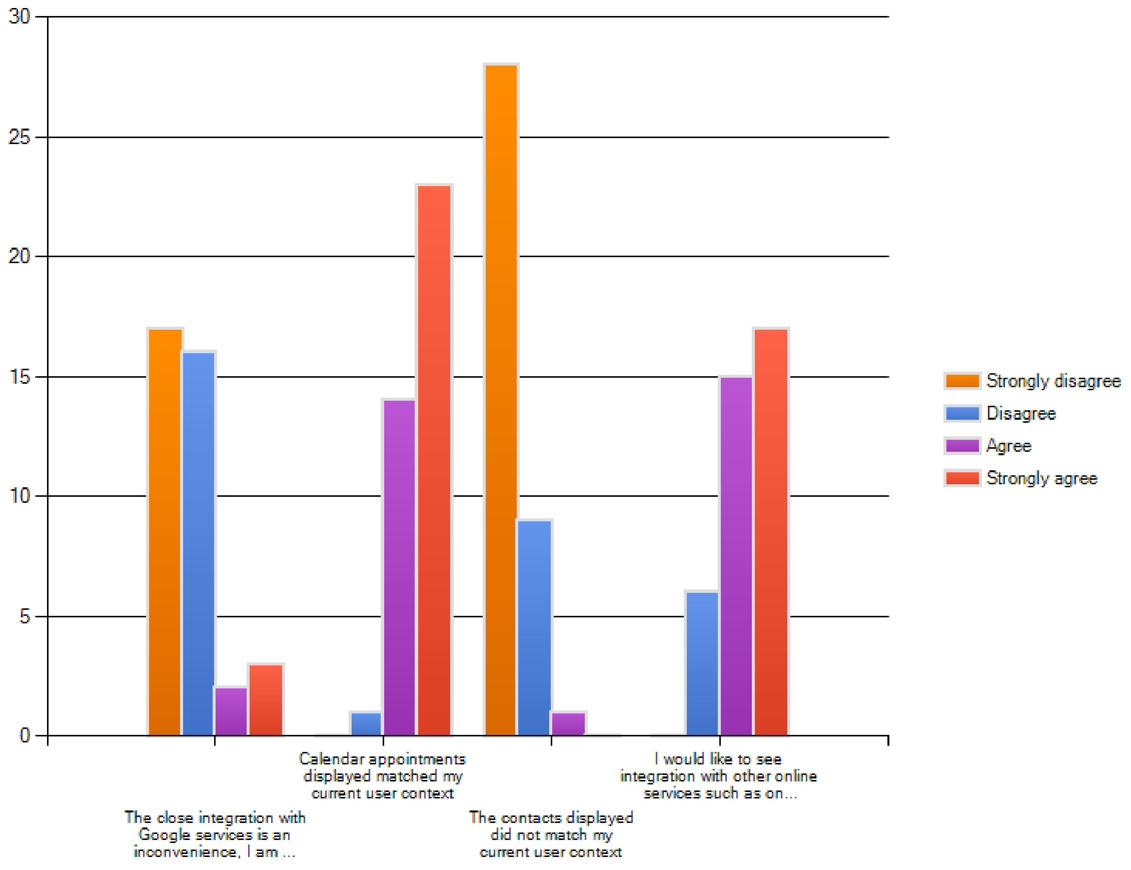
Figure 8: Statements regarding web application



Context-awareness

In terms of context-aware information the participants were asked to take a stand in respect of four statements, with results shown below (Figure 9). For the first statement (S10), although a clear majority supported this assertion (33/38), opinions are somewhat spread and this answer is not statistically significant. For the next two questions a very positive bias is shown, indicating correctly computed context-awareness and correct presentation to the users. Again for statement 14, users are eager to see more cloud-based services and integration.

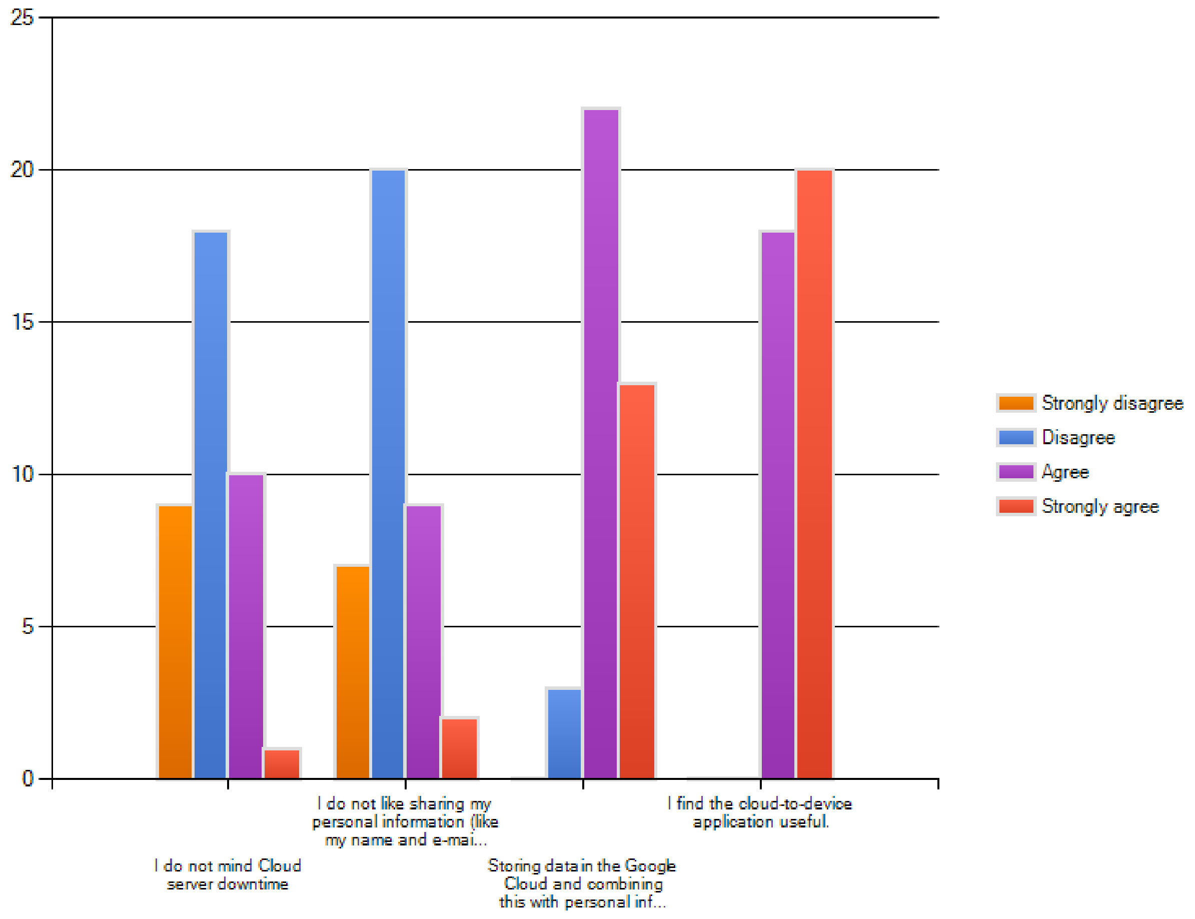
Figure 9: Statement regarding context-awareness



Cloud computing

When inspecting results from the cloud-computing section, results are mixed and differences in opinions do occur. For statements 14 and 15 the results are not statistically significant, but they indicate a mixed attitude towards cloud vulnerability and cloud data storage. The two statements with statistically significant results, statement 16 and 17, participants find storage of data in the cloud and using this as part of the data foundation for the application a useful feature and are positive towards it. Their answers also suggest a fondness for push based application configuration.

Figure 10: Statement regarding cloud-computing



RELATED WORK AND DISCUSSION

From the literature we point at the ability for modern applications to adapt to their environment as a central feature (Dey and Abowd, 1999). Edwards (2005) argued that such tailoring of data and sharing of contextual information would improve user interaction and eliminate manual tasks. Results from the user evaluation support this. The users find it both attractive as well as have positive attitudes towards automation of tasks such as push updates of information by tailoring the interface. This work has further elaborated on context-aware integration and shown how it is possible to arrange interplay between on device context-aware information, such as sensors, and cloud-based context-aware information such as calendar data, contacts and applications building upon suggestions for further research on adaptive cloud behavior as identified by Christensen (2009) and Mei et al. (2008).

Barkhuus and Dey (2003) conducted a study to examine the effects of context on user's control over mobile applications. The study defined three level of interactivity between users and mobile devices including personalization, passive context-awareness and active context-awareness. User preferences were then studied according to these three levels. The study showed that in the case of passive and active context-awareness scenarios, users felt less in control of their mobile applications. But the overall conclusion was that users were willing to compromise on losing some control if they could get some useful reward in return.#

Wei and Chan (2007) incorporated a decade of work on context-awareness and investigated the matter further. They presented three characteristics of context-aware applications:

- Context is applications specific
- Context is external to applications
- Context can be used to change behaviour, data or structures

In their work, they highlighted how context, as a source of information, represented by these three characteristics, could be used to adapt applications. They explored and categorized the information into physical context, computing contexts and user contexts (Wei and Chan, 2007). The goal of this information is to create a better experience for the user, and the way context-aware information is adopted together with the time of the adoption are major issues to be investigated. The adaptation may happen with data, behaviour or application structures. They argued that the more fundamental the adaptation is, e.g. changing structures, as well as the later the adaptation is, e.g. runtime, the harder it would be to achieve the implementation. The benefits though, might be increased, as the changes are more fundamental. Satyanarayanan (2011) exemplified context-aware attributes as: physical factors (location, body heat and heart rate), personal records and behavioural patterns. He stated that the real issue was how to exploit this information and how to deal with all the different representations of context.

To register the tags the standard Google Calendar and Contacts web-interface was used. Such a tight integration with the Google services and exposure of private information was not regarded as a negative issue. As shown in the results, most of the users surveyed disagreed that this was an inconvenience. This perception makes room for further integration with Google services in future research, where, amongst them, the Google+ platform will be particularly interesting as this may bring opportunities for integrating the social aspect and possibly merge context-awareness with social networks.

Sensors are an important source of information input in any real world context and several previous research contributions look into this topic. The work presented in this paper follows in the footsteps of research such as that of Parviainen et al. (2006), and extends sensor integration to a new level. By taking advantage of the rich hardware available on modern smartphones, the developed application is able to have tighter and more comprehensively integrated sensors in the solution. From user evaluation one can learn that although sensor integration as a source for context-awareness is well received, there is still research to do. In particular this has to do with the fact to what extent what thresholds should be used for sensor activation and deactivation. We have shown that it is feasible to implement sensors and extend their context-aware influence by having them cooperate with cloud-based services. Further research should investigate sensor thresholds and additionally see if there are differences in people's perceptions of different sensors.

CONCLUSION

Our research has added a new and novel contribution to the area of context-awareness. We have proposed and demonstrated principles in implemented applications, whereby context-aware information is harvested from several dimensions to build a rich foundation on which to base our algorithms for context-aware computation on. Furthermore, we have exploited and combined this with the area of cloud computing technology to create a new user experience and a new way to invoke control over user's phone. Through a developed application suite, we have shown the feasibility of such an approach, reinforced by a generally positive user evaluation. Moreover, we believe our solution, incorporating remote and automatically configuration of Android phone advances the research area of context-aware information. Future research should continue to innovate and expand the notion of context-awareness enabling further automatically adaptation and behaviour altering in accordance with implicit user needs.

REFERENCES

Binnig, C., Kossmann, D., Kraska, T. & Loesing, S. 2009. How is the weather tomorrow?: towards a benchmark for the cloud. Proceedings of the Second International Workshop on Testing Database Systems. Providence, Rhode Island: ACM.

Christensen, J. H. 2009. Using RESTful web-services and cloud computing to create next generation mobile applications. Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications. Orlando, Florida, USA: ACM.

Dey, A. & Abowd, G. D. Towards a Better Understanding of Context and Context-Awareness. 1st international symposium on Handheld and Ubiquitous Computing, 1999.

Edwards, W. K. 2005. Putting computing in context: An infrastructure to support extensible context-enhanced collaborative applications. ACM Transactions on Computer-Human Interaction (TOCHI), 12, 446-474.

Mcmurtrey, K. Defining the Out-of-the-Box experience: A case study. Annual conference Society for Technical Communication, 2001.

Mei, L., Chan, W. K. & Tse, T. H. A Tale of Clouds: Paradigm Comparisons and Some Thoughts on Research Issues. Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference, 2008. IEEE Computer Society, 464-469.

Mell, P., Grance, T. (2011). The NIST Definition of Cloud Computing. National Institute of Standards and Technology, Special Publication 800-145

Parviainen, M., Pirinen, T. & Pertilä, P. 2006. A Speaker Localization System for Lecture Room Environment. Machine Learning for Multimodal Interaction, 225-235.

Baldauf, M., Dustdar, S., Rosenberg, F. 2007. A Survey on Context-aware Systems. Int. Journal of Ad Hoc and Ubiquitous Computing, Vol. 2(4): 263–277.

Kapitsaki, G.M., Prezerakos, G. N., Tselikas, N.D., Venieris, I.S., 2009. Context-aware Service Engineering: A Survey. Journal of Systems and Software, Vol. 82 (8) 1285-1297

Younas, M., Awan, I. 2013. Mobility Management scheme for Context-aware Transactions in Pervasive and Mobile Cyberspace. IEEE Transactions on Industrial Electronics, Vol. 60(3), 1108-1115

Barkhuus, M., Dey, A.K. 2003. Is Context-Aware Computing Taking Control away from the User? Three Levels of Interactivity Examined. Proc. of the 5th International Conference on Ubiquitous Computing (UbiComp), Seattle, WA, USA, October 12-15, 2003, LNCS 2864 Springer, ISBN 3-540-20301-X, 149-156

Satyanarayanan, M. 2011. Mobile computing: the next decade. SIGMOBILE Mobile Computing Communication Rev., 15, 2-10

Wei, E. & Chan, A. 2007. Towards Context-Awareness in Ubiquitous Computing. International Conference on Embedded and Ubiquitous Computing (EUC 2007), Taipei, Taiwan, December 2007, LNCS Springer, Vol. 4808, 706-717

Google. 2012. What Is Google App Engine? [Online]. Available: http://code.google.com/appengine/docs/whatisgoogleappengine.html