EVALUATION ISSUES FOR UBIQUITOUS COMPUTING

Jung-Hyun Bae, Soon-il Cha, Seok-Kyu Shin

Telecommunications Technology Association, 267-2, Seohyun-dong, Bundang-gu, Seongam-city, Gyeonggi-do, Korea

Keywords: Ubiquitous Computing (UC), Evaluation, Quality Characteristic.

Abstract: One of the main problem with the evaluation of ubiquitous computing (UC) is that we do not have standards guidelines, or metrics, even though there have been many researches on the individual technical aspects of ubiquitous systems. UC systems differ from standard applications in many ways. Therefore, it is very hard to apply the current working standards or metrics to the evaluation of UC system, which leads us to develop a desirable way to deal with this. This paper suggests the evaluation issues of service and system software to provide ubiquitous computing. Also, we investigate how equally the evaluation scopes we have suggested and the quality characteristics could be matched with the current technological feasibility of the ubiquitous computing systems, so we can see what benefits and the deficiencies of the proposed evaluation scopes would be. Two case studies are considered and these can be seen as the most-likely applications in the current ubiquitous computing environment.

1 INTRODUCTION

For decades, the improvement of communications technologies has brought computing environment evolved into human from machine. This is called a ubiquitous computing environment, initially introduced within the computer systems world by Weiser in the early 1990s (Weiser, 2002), comprehends the ability to exist everywhere at the same time. Ubiquitous environment consists of innumerable number of computing devices embedded in almost everything around us, platforms and networks that interconnect them, and user devices that make use of and act on the available information. The system software in this domain, it is a key issue to provide satisfied UC services to users without intrusion among heterogeneous devices with different capabilities and protocols. The ubiquitous computing has environmental problems such as wireless channel fading and mobility. As these problems have possibilities to affect the performance of a network and application, it is often proposed that quality-of-service (QoS) be handled at the ubiquitous system software level (Tokunaga et al, 2004).

This paper proposes evaluation issues of services and the system software running on the ubiquitous computing environment. And we investigate the feasibility for application of the proposed evaluation scopes through considering two case studies.

2 UBIQUITOUS COMPUTING ENVIRONMENT

A general ubiquitous computing system environment has four entities; User, I/O artefact, Service, Ubiquitous System Software as shown in Figure 1 (Cha et al., 2005).

When the receptive events are triggered by users' activities, sensors or some physical input artefacts carry out the operations of reception and integration of the user's activities. Examples of some physical input artefacts are switches mounted to a grip, trackballs, and mobile phone, gesture and speech recognition or context awareness facilities. The receptive function accepts the user's activities and act upon it, while the integrated function takes into account the states of the several sensors as well as the input artefacts at the same time, delivering the information to the ubiquitous service. Once the sensors or the physical input artefacts locate a detectable event, the ubiquitous services act upon it to process the integrated event.

Most of the ubiquitous application modules receive a lot of data from other input artefacts or theirs and interpret this data to fit into the context of the user's activities. The context awareness in the ubiquitous services is of great importance (Asim, 2004). A very significant part of the context is to know or be able to infer the user's intent. For instance, Global

Bae J., Cha S. and Shin S. (2006).
EVALUATION ISSUES FOR UBIQUITOUS COMPUTING.
In *Proceedings of the International Conference on Wireless Information Networks and Systems*, pages 317-320

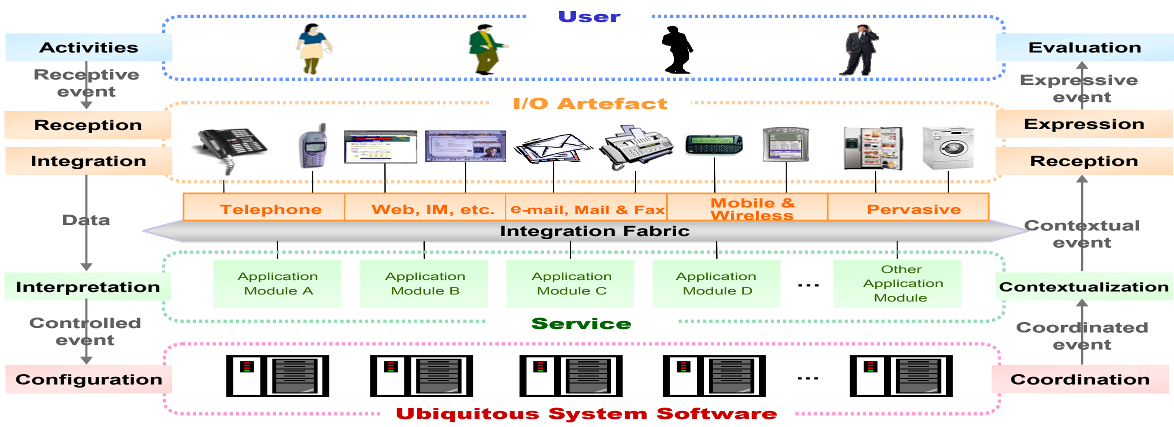

Figure 1: Interaction in UC System Environment.

Positioning Systems (GPSs) give us location context. Some good sensing networks provide environmental context, and clever service design can support some levels of intent inference, i.e., situational context.

Ubiquitous system software controls all the data processing for these individual entities in the ubiquitous system and coordinates the contextual-lization between services so that the services can make intelligent decisions on how to interpret events.

The output artefacts formulate expressive functions upon contextual events triggered from the ubiquitous service. Finally, users evaluate the expressive events from output artefacts e.g., large-wall-mounted display, lights, sound or even hap-tic interfaces, in the system.

In a UC environment, service not only provides configuration of integrated event (or data from the sensors or physical devices) but also contextualization with regard to the information that the ubiquitous computing system has to provide. The system should provide for the correct translation between data types and representations of the outputs generated by one service that are to be used as the inputs to another. Therefore, service and system software can usually be chained together to form a complicated task.

3 EVALUATION ISSUES FOR SERVICE AND SYSTEM SOFTWARE

Under the UC environment, we must consider the one person-to-many computing interaction. This paper figures out main function and quality characteristics for service and system software entities.

Table 1 and Table 2 delineate evaluation scope with the checklist examples and metrics for each entity.

3.1 Service

The UC service refers to the applications that perform computation or action on behalf of users in a ubiquitous computing environment.

We have found several issues crucially important for the successful deployment of services in the UC environment: Functionality, service quality, smartness and security. These are the key characteristics to make UC possible.

Functionality

The ubiquitous computing service has the features necessary to support the requirements. The general purpose of functionality testing is to verify if the product performs as expected and documented. Developers creating a new product start from a functional specification, which describes the product's capabilities and limitations. Software functionality test engineers utilise this specification as a guideline for expected product response. Tasks are exercised to test specific features or functions, and the results of the tasks are verified to be in *compliance*, *suitability* and *accuracy* with the expected response.

Service Quality

For varying application computation and communication demands and for a varying quality offered by the ubiquitous computing environments, the ubiquitous system has to provide multiple grades of service quality and maintain a number of agreed-upon or negotiated service qualities. Technically, quality of service (QoS) can be guaranteed through bandwidth, ratio frequency and response time requirements. In terms of evaluation objectives, several characteristics of service quality should be considered as followings: *efficiency*, *resource availability*, *reliability*, *maturity*, *fault tolerance*, *fault recovery, maintainability* and *adaptability*.

Smartness

Smartness is often referring to Smart house or smart room where small sensors, such embedded processors are used to detect their surroundings and equip human with both information processing and communication capabilities. A smart house is a truly interactive house using the latest information and communication technology to link all the mechanical and digital devices available today. Humans could communicate and cooperate with other smart objects and, theoretically, access all sorts of Internet resources. Objects and appliances could thus react and operate in a context-sensitive manner and appear to be smart. Five aspects of smartness should be considered for evaluation purpose: *location awareness*, *environment awareness*, *situation awareness*, *user recognition* and *personalization*.

Table 1: Evaluation Scope of the UC Service.

Criteria	Characteristic	Checklist Example	Metric(s)
Functionality	Compliance	How compliant is the functionality of the product to applicable regulations, standards and conventions?	Functional compliance ratio X = 1 –A/B, where A = No. of functionality compliance items specified that have not been implemented, and B = Total No. of functionality compliance items specified
	Suitability	How adequate are the evaluated functions?	Function Adequacy X = A / B, where A = No. of functions in which problems are detected, and B = Total No. of functions
	Accuracy	Are the differences between the actual and reasonably expected results acceptable?	Accuracy to expectation Ratio = A / T, where A = No. of cases encountered by the users with a difference against the reasonably expected results beyond allowable, and T = Operation time
Service Quality	Efficiency	What is the absolute limit on memory required in fulfilling a function?	Maximum memory Utilisation X = Amax /Rmax, where Amax = MAX(Ai), (for i=1 to N), Rmax = required maximum No. of memory related error messages from 1st to i-th evaluation, and N = No. of evaluations
	Resource availability	What is the availability of the system?	Availability A = MTTF/(MTTF +MTTR), where MTTF is the mean time to failure. MTTR is the mean time to repair.
	Reliability	How to measure system reliability?	Reliability R(u) = exp(-uλ(x)) = exp(-u/MTTF). Here u is the projected execution time in the future, x is a variable of integration, and λ(x) is the failure rate.
	Maturity	How many faults were detected in this product?	Fault Detection Rate = A/B, where A = absolute number of faults detected in this product, and B = number of estimated faults to be detected (using past history or modelling techniques)
	Fault Tolerance	How many fault patterns were brought under control to avoid critical and serious failures?	Fault avoidance rate = A /B, where A = Number of fault patterns having avoidance, and B = Number of fault patterns to be considered
	Fault recovery	How capable is the product in restoring itself after abnormal event or at request?	Recovery Rate =A /B, where A = No. of implemented recovery requirements confirmed, and B = No. of recovery requirements in the specification
	Maintainability	Can user identify specific operation which caused failure?	Failure Analysis Capability X = 1- A /B, where A =No. of failures of which causes are still not found, and B = Total No. of registered failures
	Adaptability	Can user or maintainer easily adapt software to data sets in new environment?	Adaptability of data structure X = A /B, where A = No. of data which are operable but are not observed due to incomplete operations caused by adaptation limitations, and B = No. of data which are expected to be operable in the environment to which the software is adapted
Smartness	Location awareness	How accurate can the ubiquitous computing system sense the user location?	Location Sensitivity = confidence intervals $(\alpha\pm \beta\%)$ in terms of the real location of the user
	Environment awareness	Is the system capable of providing environment-aware, such as location-aware and orientation-aware sensor data?	Environment Sensitivity = confidence intervals $(\alpha\pm \beta\%)$ in terms of the real environment of the user
	Situation awareness	Is the system capable of providing environment-aware, such as location-aware and orientation-aware sensor data?	Situation Sensitivity = confidence intervals $(\alpha\pm \beta\%)$ in terms of the real situation of the user
	User recognition	Can the ubiquitous computing system automatically identify the user?	Automatic Identification Rate = A /B, where A = No. of cases the ubiquitous computing system identifies a user correctly, and B = Total No. of cases the ubiquitous computing system needs to identify a user
	Personalization	Is the ubiquitous computing system capable of providing personalised information according to user preferences?	Personalisation Rate X = A /B, where A = No. of times the ubiquitous computing system provides correct personalised information to a specific user, and B = Total No. of times the ubiquitous computing system need to provide correct customised functions to a specific user
Security	Authentication	How complete is the audit trail concerning the user access to the system and data?	Access Auditability X = A /B, where A = No. of "user accesses to the system and data" recorded in the access history database, and B = No. of "user accesses to the system and data" done during evaluation
	Privacy	Is data encrypted to protect the wearer's privacy? - To prevent the wearer's identity being discovered - To prevent the wearer's personal details(sensor data) from being discovered - To Provide encryption?	Data Intrusion Prevention Metrics: - X = 1- A/ N, where A = No. of times that a major data intrusion event occurred, and N = No. of test cases tried to cause data intrusion event - Y = 1- B/N, where B = No. of times that a minor data intrusion even occurred - Z = A /T or B / T, where T = period of operation time (during operation testing)

Security

The security of UC service refers to establish mutual trust between infrastructure and device in a manner that is minimally intrusive. Security can be implemented in heterogeneous components such as firewalls, different computer operating systems and multiple databases. Authentication is one of the most important characteristics of ubiquitous computing security. Authentication provides confirmation of user access rights and privileges to the information to be retrieved.

3.2 System Software

UC device hardware is not simply limited to handheld devices. There is also a large amount of fixed-infrastructure hardware that may be incorporated into a system. The software that runs the hardware devices, and coordinates the usage of the physical resources is of paramount importance. The UC systems software and networks have two orthogonal aspects: configuration and coordination.

Configuration

The systems software and networks used by a UC system need to be able to be adequately configured so that the ubiquitous computing system can effectively use the hardware. From a purely configurational point of view, the most important aspects of configuration management are: *scalability* and *extensibility*.

Coordination

Coordination describes the system software's ability to coordinate a user request operating across the ubiquitous computing system. We sub-divide this category into *resource coordination*, *mobility*, *reliability* and *distributed systems support*.

Table 2: Evaluation Scope of the UC System Software.

Criteria	Characteristic	Checklist Example	Metric(s)
Configuration	Scalability	Does the ubiquitous computing system support the ability to add new hardware?	1-(A/B) where A is the response time of the system after new hardware is added and B is the response time of the system before the new hardware was added
		Is there a list of drivers or APIs that can be used by developers or technical people to use a new piece of hardware with the ubiquitous computing system?	A/B where A is the number of jobs (queries, service invocations) that are waiting to be executed and B is the number of jobs that are concurrently being executed on average (lower is better).
	Extensibility	Is there a programmer API that can be used to extend the system?	A/B where A is the number of software and hardware systems that can be upgraded in the field and B is the total number of hardware and software systems in the environment that could need upgrading
		Can programmers interrogate the services or other hardware used by the system?	A/B where A is the number of software systems that have technical support available from the Vendor and B is the total number of software systems in the environment
Coordination	Resource coordination	How well does the systems software control resource sharing on the local hardware	1-(A/B) where A is the response time of the system after a new service is invoked and B is the response time of the system before the service was invoked.
	Mobility	Is the resource monitoring aspect of the systems software sophisticated enough to judge whether a program or code would be more efficient if located elsewhere?	1-(A/B) where A is the time for the system to locate a re-located piece of hardware and B is the time to locate the hardware before moving
		Is there enough information available on remote resources to make a good judgement on where to locate code/data	1-(A/B) where A is the time for the system to access a re-located piece of hardware (after the system is quiescent) and B is the time to access the hardware before moving
	Reliability	Does the system continue working in the presence of a variety of faults?	What is the availability ('up-time') of the hardware? 1-(A/B) where A is the time that the hardware has been unavailable due to fault(s) and B is the sum of the time the hardware has been operating and to has been unavailable
		What technical support is available for the system software or hardware?	What is the availability ('up-time') of the systems software? 1-(A/B) where A is the time that the systems software has been unavailable due to fault(s) and B is the sum of the time the systems software has been operating and to has been unavailable.
	Distributed systems support	Is the systems software a 'distributed/network operating system' or a 'middleware layer'?	1-A/B where A is the number of machines that have been booted in the environment and users are still able to submit new jobs (and have them complete) and B is the total number of machines in the environment
		To what level are the distributed physical resources inter-dependent?	1-A/B where A is the number of machines still operating (in the presence of other machines being unavailable) and users are still able to submit new jobs (and have them complete) and B is the total number of machines in the environment

4 CASE STUDY

We consider two case studies and investigate the propriety of the proposed evaluation scopes for UC service and ubiquitous system software.

4.1 Mobile Banking Service

The main applications of the ubiquitous service would be via mobile phones. SK telecom and Telecom NZ introduced the mobile banking service in 2004. The customer can use the service anywhere without distorting or blocking the signal. Service provider's global network allows the mobile banking to be sued around the world using their roaming service (Maintainability – Service Quality). One of the major banks in New Zealand, ASB bank has introduced mobile banking service, taking security seriously and offering the security of a secure 128 bit encrypted connection. Furthermore, the customers are required to enter their personal access code and passwords to sign on the system (Security – Service). They also provide some emergency situation control such as battery discharges while doing banking service. All of the banking transactions the customers made before their phone battery discharged should have done through. Once battery is recharged, customers can check the balances and the record of the last transactions the customers made. Alternatively, because all the online service are real time, the customers can also check the balanced and last transactions through internet banking or telephone banking service (Reliability – Service Quality and Coordination). In addition, as the customers' accounts will be displayed on the screen of the mobile phone, so the customer can access their accounts quickly without fuss when they are on the move (Functionality – Service). This service is also designed to cope with a large number of users and has a back-up system (Scalability – Configuration).

4.2 Location-Awareness Service

As another application of the ubiquitous service via mobile phones, Vodafone Germany (2000) introduced the first location awareness services based on the WAP portal in the world. All location-based services offered so far to mobile phone users are always pull-services, say; the user has to explicitly request information relevant to her current location. This is particularly annoying if the user always has to establish a WAP connection before he can use such services. It takes a lot of time, which limits the values of the service to the user, and it is also very costly as the user usually has to pay for the online time. Thus, this service should be very easy to use, therefore must offer a simple, yet powerful interface, be robust, and error-tolerant, covering the reasonable service region (Reliability – Service Quality and Coordination, Resource availability – Service Quality). The location-awareness service must also be secure and private to the users. The user should be able to turn on/off the service to actively confirm her wish for localization for privacy reasons (Privacy – Service). The most important technical concern of this service is the high accuracy location enabler across all environments (Location awareness – Smartness). However, in order to realize the proper location awareness service, information has to be automatically delivered to the user (Functionality, Adaptability – Service). The system architecture for tracking location should also not be restricted to or depend on specific kinds of trackers. It must allow adding new trackers (perhaps even newly developed trackers) and including them easily in the overall architecture (Extensibility, Scalability – Configuration).

5 CONCLUSION

This paper outlines the issues to be considered in terms of UC service and system software evaluation. We also apply the proposed evaluation scopes to two case studies. However, these are not a full evaluation scope but some important things. We expect that these issues can be the beginning point to make the systematic UC system evaluation model along with the present software evaluation model such as ISO/IEC 9126 or 12119. Our work raises the need of framework development and opens a new discussion for the UC evaluation.

REFERENCES

M. Weiser, 2002. The Computer for the 21st Century. *IEEE Pervasive Computing*, 1 (1), pp.18-25.

Tokunaga. E, et al, Aug. 2004. A middleware infrastructure for building mixed reality applications in ubiquitous computing environments, *MOBIQUITOUS 2004*, pp.382-391.

S. I Cha, J. H Bae, et al, Jan.2005. Evaluation issues of ubiquitous computing systems, *APIS 2005*, pp.637-640.

Asim. S, Daniel. S, Aug. 2004. Application design for wearable and context aware computers, *IEEE Pervasive Computing*, pp.20-29.