Checklist to Characterize Ubiquitous Software Projects

Rodrigo O. Spínola, Jobson Massollar, Guilherme H. Travassos

COPPE / UFRJ – System Engineering and Computer Science Department
Caixa Postal 68.511 – 21.945-970 – Rio de Janeiro – RJ – Brasil

{ros,jobson,ght}@cos.ufrj.br

Resumo. *OBJETIVO: Identificar as características de computação ubíqua e propor um checklist para caracterização de projetos de software segundo as características de ubiqüidade. MÉTODO: Executar revisões sistemáticas para entender: 1) computação ubíqua, 2) suas principais características e 3) seus fatores específicos. Depois disto, definir e avaliar o checklist proposto. RESULTADOS: 116 artigos foram analisados resultando em uma definição para computação ubíqua e na identificação de 10 características (associadas a 113 fatores funcionais e 45 restritivos) que foram organizadas em um checklist para caracterização de projetos de software ubíquos. Oito projetos de software ubíquos foram utilizados para avaliar o checklist. Nenhum deles pôde ser caracterizado como 100% ubíquo. CONCLUSÃO: A definição atual de computação ubíqua possui conceitos (características e fatores) que permitem a caracterização de projetos de software ubíquos. Entretanto, deve ser investido esforço adicional de pesquisa para entender como as características de ubiqüidade podem influenciar na engenharia de software.*

Abstract. *OBJECTIVE: To find ubiquitous computing characteristics and propose a checklist for characterizing software projects regarding ubiquity. METHOD: To run systematic reviews to understand: 1) ubiquitous computing, 2) its main characteristics, and 3) its specific factors. After that, to configure and evaluate the checklist. RESULTS: 116 papers were analyzed resulting in the current definition for ubiquitous computing and the identification of 10 characteristics (associated with 123 functional and 45 restrictive factors) that have been organized into a checklist to characterize ubiquitous software projects. Eight self-nominated ubiquitous software projects were used to evaluate the checklist. None of them could be characterized as 100% ubiquitous. CONCLUSION: The current definition of ubiquitous computing embraces concepts (characteristics and factors) that allow the characterization of software projects regarding ubiquity. However, additional research effort must be invested towards the understanding of ubiquitous characteristics influence in the software engineering.*

1. Introduction

Many recent software projects have been supporting requirements regarding the usual perception of ubiquitous computing. These demands usually require device independence, omnipresence, alternative interfaces that changes the usual computer screen and mouse paradigm, adaptation to the surrounding environment and so on, to create a scenario where ubiquitous computing can be seen as a new information access paradigm. According to Weiser (1991), computers should be embedded into the environment in such way that their use becomes natural and transparent.

This prospective scenario can represent new research challenges in many areas (not intending to be restrictive) like computer network, signal processing, optimization, and artificial intelligence. Particularly, from the point of view of software engineering, these challenges can effectively be observed on the development of methodologies, software processes, testing approaches, and quality assurance techniques. That is what we have observed when dealing with some innovative software projects regarding e-science in Brazil. Our software engineering group faced some of these issues when it was requested to plan the development of a new project to support experimentation in large scale by multidisciplinary teams, where one of the requirements explicitly mentioned the characteristic of ubiquity.

From that moment, we have identified the need of understanding the impact of the ubiquitous characteristic on many phases of the software project development. For instance:

- What (new) software engineering techniques are necessary to deal with the ubiquity characteristic of software?
- Do exist testing approaches able to ensure the quality of ubiquitous software projects? What could be the extensions/adaptations/new approaches needed?
- What are the risks associated with ubiquitous software projects?
- What quality characteristics software engineers should have in mind for ubiquitous software projects?

The challenges imposed by the requirements of this new software category drove us to think about the need of a complementary software engineering body of knowledge, amending the one that has started to be built to the planning of customary software projects. Otherwise, software engineers will have to rely on their own experiences and capacity to tailor their knowledge to be used as an ad-hoc development approach for this new category of software, what has been observed for conventional software development not to be a good approach.

Therefore, the process of acquiring information to reduce the risks concerned with the development of ubiquitous software projects begins with the identification of the impact of ubiquitous characteristic on software projects. This way, it would be relevant to characterize the level of adherence of a software project according to a classification criteria supported by ubiquitous features.

Based on this context, we believe that an essential initiative could be to look for some way that allows characterizing a software project based on the characteristics that identify it as ubiquitous. Therefore, we have started some research intending to support the finding these characteristics aiming at the description of a checklist to characterize software projects accordingly their adherence level regarding ubiquitous computing.

To develop such checklist, a fundamental task is concerned with the definition of ubiquitous computing and its main characteristics, since all the software projects should be classified according to the same perspective and the term ubiquitous has been used in a broad and, sometimes, ambiguous way. Besides that, although ubiquitous computing has been investigated into different contexts, to our knowledge there has been no centralized research effort in properly define the characteristics and factors of software and systems focused on ubiquity and how they could influence software processes.

Here, it is important to explain the choice of the terms characteristics and factors used in this paper. According to the Cambridge Dictionary:

- Characteristic is a "typical or noticeable quality of someone or something". Thus, we can consider characteristic as a typical or noticeable quality of ubiquitous software;
- Factor is a "fact or situation that influences the result of something". In the context of this work, we can consider factor as a fact or situation that influences a ubiquitous software characteristic.

In order to reach a solid scientific level, we decided to execute systematic literature reviews [Kitchenham 2004, Biolchini et al. 2005] rather than *ad-hoc* ones. This approach, unlike the *ad-hoc* approach, is based on the scientific method and follows an explicit process for conducting the review based on a formal research protocol.

We have conducted two systematic reviews. The first one aimed at the identification of the current ubiquitous computing definition, identifying where it is currently being used and which are its main characteristics. The results allow us to say that, besides its definition, ubiquitous computing can be represented by 10 different characteristics. However, these characteristics are still described in so high abstraction level that it is not possible to use them directly for classifying software projects. Therefore, a second systematic review was accomplished aiming at the identification of functional and restrictive factors associated with these ubiquitous characteristics. These factors are concerned with the facts or situations related to functional and non-functional requirements respectively.

About 123 functional factors and 45 restrictive factors have been identified as important when considering ubiquitous software projects. These factors were distributed among the ubiquitous characteristics, improving the characterization criteria used by the checklist. To exemplify, we have applied such checklist to characterize 8 different ubiquitous software applications identified during the execution of the first systematic review and that not have been used to compose the checklist, what can give some directions on how to use it for a larger population of software projects.

Some initial results from the systematic reviews have been reported in [Spínola et al. 2006]. However, new research results have been obtained since then. Therefore, this paper brings some of these results represented by: (1) presenting in detail the systematic reviews protocols; (2) presenting the consolidated results of characteristics and factors definition; (3) discussing how ubiquitous characteristics and their factors were organized to create the proposed checklist; (4) exemplifying the checklist usage to characterize eight ubiquitous software projects, and; (5) reporting about the current research status and the further steps planned to improve its results.

This paper is organized in seven sections, including this Introduction. In section 2, an introduction to systematic review is made. In section 3, the definition and characteristics of the ubiquity are presented. In section 4, the results of a detail analysis for each ubiquitous characteristic are presented. In Section 5, we propose an approach to classify applications considering their ubiquitous adherence level. In section 6 we present some results obtained with the classification approach and some insights about the distance between the states of the art and practice in ubiquitous computing. Finally,

in section 7 we present the main contributions of this paper and future perspectives of this research project.

2. Systematic Review

During the study of a new knowledge area, researchers usually conduct a literature review to identify publications related to a specific subject. However, this kind of review does not use a systematic approach and does not offer any kind of support to avoid bias during the selection of the publications that will be analyzed. A way to systematize the identification and analysis of publications is using systematic reviews [Kitchenham 2004].

A systematic review is a mean of identifying, evaluating and interpreting all available research relevant to a particular research question, topic area, or phenomenon of interest [Kitchenham 2004]. In comparison with traditional reviews of literature, systematic review requests higher rigidity for its accomplishment. Its results tend to be more reliable because it makes use of a rigorous methodology that is susceptible to auditing and replication. Besides that, there are more specific reasons to justify the systematic review use [Kitchenham 2004]: (1) to summarize, for instance, some existing evidences on a theory or technology, (2) to identify open issues for the research, making possible the identification of areas where more investigations should be accomplished, and (3) to provide a basis for new research activities.

This approach is based on a specific sequence of activities, according to a research protocol previously defined. Thus, for each systematic review execution, the following process can be used [Biolchini et al. 2005]:

- Review planning: the research's purpose has to be specified by means of defining what will be searched, the sources where the search will be carried through and the criteria used to select the studies. At the end of this step, a version of the protocol has to be created and the feasibility of the review has to be evaluated.
- Review execution: the studies related to the research goals and that satisfy the selection criteria are identified. This identification is accomplished using key words based searches defined on the protocol.
- Result analysis: the identified studies are analyzed to answer the research questions.

In this work, we defined two complementary research protocols whose results are presented in the next two sections.

3. Ubiquitous Computing: Definition and Characteristics

Weiser (1991) defines ubiquitous computing as being the use of computer through its availability in the surrounding environment, making it effectively invisible to the user. That is, the computer is integrated in the environment in such a way that its use becomes not intrusive. This definition set the origin of the term Ubiquitous Computing and, although it is important for presenting a new computing paradigm, it is not complete at all. This lack of completeness reflects how this proposal was innovative and technological limitations found at that time.

In this context, this section presents the reached results from the first systematic review execution. The goals of this review were to answer the following questions:

- Q0: What is ubiquitous computing?
- Q1: How ubiquitous computing is actually being presented?
- Q2: What characteristics do define applications for ubiquitous computing?

To accomplish this systematic review, it was elaborated a research protocol. The items below define the main characteristics of this protocol:

- **Keywords**: ubiquitous computing, pervasive computing, ubiquitous application, ubiquitous system, ubiquitous software, pervasive application, pervasive system, pervasive software, feature, requirement, characteristic, definition, characterization, and concept.
- **Paper sources**: IEEE Portal, ACM Digital Library, INSPEC, and EI COMPENDEX. These digital libraries have been chosen by convenience, since they were fully available for the researchers.
- **Examples of search strings** (Q0 only): the search strings were defined based on syntax used by the IEEE and ACM search machines, thus, it is important to know some keywords: metadata means that the term will be looked for on the whole paper; abstract means that the term will be looked for only on the paper abstract;

  IEEE:
  ((('pervasive computing' <or> 'ubiquitous computing')<in>metadata) <and> ((definition <or> concept <or> characterization)<in>metadata)) <and> (pyr >= 1991 <and> pyr <= 2005)

  The filter (pyr >= 1991 <and> pyr <= 2005) because the seminal paper was published at 1991 and this literature review was executed at 2005.

  ACM:
  +"ubiquitous computing" abstract:concept abstract:definition abstract:characteristc
  +"pervasive computing" abstract:concept abstract:definition abstract:characteristic
- **Inclusion and exclusion criteria**: These criteria define statements that must be true for the paper be included on the set of the selected papers. The papers must: be available on the internet, be written in English, provide a ubiquitous definition (Q0 only), report current applications regarding ubiquitous computing concepts (Q1 only), report software application (applications related to supporting software are not considered) and present characteristics associated with ubiquitous systems (Q2 only).
- **Preliminary studies selection process**: each publication returned had its abstract and introduction analyzed by two researches and, based on the inclusion and exclusion criteria, they were selected or not to a more thorough analysis.

During this review, 751 scientific papers were identified and after their analysis, only 57 were selected to information extraction. Two factor contributed a lot for that great difference: (1) the inclusion/exclusion criteria defined *a priori* in the systematic protocol where applied to define which papers would be selected, and; (2) an unexpected behavior of some search engines inflated the total number of identified papers. For instance, a recurrent problem was the inclusion of papers with the keywords present only in the footnotes.

Within the 57 papers, eight describe concrete ubiquitous applications [Tahti et al 2004] [Kindberg et al 2000] [Ali et al 2004] [Bossen and Jorgensen 2004] [Hatala et al.

2005] [Lee and Chung 2004] [Zhou et al 2005] [Joel et al 2004]. The definition and characteristics of ubiquity obtained as result of the systematic review execution are presented below.

Ubiquitous computing definition: ubiquitous computing is present when computational services or facilities become available to the people in such a way that computer is no longer a visible or essential tool to access these services or facilities. That is, services or facilities are accessed at any time or place, transparently, through the use of common devices. To make it happens it is necessary that systems that make part of this scenario take into consideration the following project issues: service omnipresence, invisibility, context sensitivity, adaptable behavior or task dynamism, capture of experiences, service discovery, function composition, spontaneous interoperability, device heterogeneity and fault tolerance. We called these project issues ubiquitous computing characteristics.

Ubiquitous systems definition: we can notice that ubiquitous computing definition represents the "philosophy" of this new computing paradigm. This way, it defines the ideal conditions where we can access computational resources in a ubiquitous way. On the other hand, ubiquitous system definition has a well-defined scope and it is strongly related to different characteristics that compose ubiquitous computing. It happens because, as ubiquity can be a property of a system, it can be achieved completely or partially. This variation is related to the fact that a particular system can implement or not the functions representing ubiquitous computing characteristics.

Ubiquitous Computing Characteristics: the definition presented by Weiser (1991), despite the fact it was the initial landmark on ubiquitous computing, presents too little about what characteristics a ubiquitous application could have. However, after the identification and analysis of the applications' characteristics, we could notice that many concepts explored by these applications have the same meaning but with different names or approaches. This way, the results obtained pointed out in the direction of a set of characteristics that can be interpreted as common ubiquitous applications characteristics. They are:

- Service omnipresence (SO): it allows users to move around with the sensation of carrying computing services with them;
- Invisibility (IN): ability of being present in objects of daily use, weakening, from user's point of view, the sensation of explicit use of a computer and enhancing the perception that objects or devices provide services or some kind of "intelligence". With that, it is possible to find proper alternatives for traditional graphical interfaces used on desktop solutions in favor of more natural ways of data input in such a way that the interface itself will be minimally perceived by the user;
- Context sensitivity (CS): ability to collect information from the environment where it is being used;
- Adaptable behavior (AB): ability of, dynamically, to adapt offered services according to the environment where it is being used, respecting its limitations;
- Experience capture (EC): ability of capturing and registering experiences for later use;

- Service discovery (SD): pro-active discovery of services according to the environment where it is being used. The application has to interact with environment and allow user to do the same, in order to find new services or information to achieve some desired target;
- Function composition (FC): ability of, based on basic services, to create a service required by the user;
- Spontaneous interoperability (SI): ability to change partners during its operation and according to its movement;
- Heterogeneity of devices (HD): provides application mobility among heterogeneous devices. That is, the application could migrate among devices and adjust itself to each of them;
- Fault tolerance (FT): ability to adapt itself when facing environment's faults (for example, on-line/off-line availability).

Although it was possible to capture a clear-cut set of characteristics, these characteristics still represent a high abstraction level, that is, it is possible to define their meaning, but it is difficult to establish their impacts on ubiquitous software projects. This way, we need to go down into a set of more concrete factors associated with each characteristic. That is the goal of the secondary study presented in the next section.

4. Functional and Restrictive Factors of the Ubiquitous Characteristics

This section presents the results obtained with the execution of the second systematic review. Its goal was to answer the question: what are the functional and restrictive factors that characterize each ubiquitous characteristic?

To accomplish this second systematic review, it was elaborated another research protocol. The items below define its main characteristics:

- **Keywords**: ubiquitous computing, pervasive computing, functional requirement, functionality, feature, characteristic, non-functional requirement, quality requirement, invisibility, context sensitivity, adaptable behavior or task dynamism, capture of experiences, service discovery, spontaneous interoperability, device heterogeneity and fault tolerance.
- **Paper sources**: IEEE Portal and ACM Digital Library. These digital libraries have been chosen by convenience, since they were fully available for the researchers.
- **Examples of search strings**:

  IEEE:
  (('pervasive computing' <or> 'ubiquitous computing') <in> metadata) <and> (('functional requirement' <or> functionality <or> feature <or> characteristic) <or> 'non-functional requirement' <or> 'quality requirement')) <in>metadata) <and> ('computer everywhere') <and> (pyr >= 1991 <and> pyr <= 2005)

  ACM:
  +"ubiquitous computing" +"functional requirement" +"computer everywhere"
  +"ubiquitous computing" +functionality +"computer everywhere"
  +"ubiquitous computing" +feature +"computer everywhere"
  +"ubiquitous computing" +characteristic +"computer everywhere"
  +"ubiquitous computing" +"non-functional requirement" +"computer everywhere"
  +"ubiquitous computing" +"quality requirement" +"computer everywhere"

- **Inclusion and exclusion criteria**: the papers must be available on the internet, the papers must be written in English and the papers must provide functional and/or restrictive factors associated with each ubiquitous characteristic.
- **Preliminary studies selection process**: each publication returned had its abstract and introduction analyzed by one research and, based on the inclusion and exclusion criteria, they were selected or not to a more thorough analysis. If some doubt arises, a second research will help on decision-making.

During this review, 599 scientific papers were identified and after its analysis, only 59 were selected for information extraction. It is important to notice that the same reasons presented on section 3 can be used to justify the great difference between these two numbers. At this point in time, we did not find any way to eliminate such bias into the search engines.

From the 59 papers selected it was possible to identify 168 factors - 123 functional and 45 restrictive (the complete set of functional and restrictive factors can be found at http://www.cos.ufrj.br/~ros/ubforms.html). Moreover, it was possible to group the factors according to their definition, associating each factor to one and only one group of factors. It is important to say that these groups were created as a consequence of the identified factors and their relationship. For example, for the "Context Sensitivity" characteristic, the factors "Contextualize obtained information" and "Store information" can be grouped on "Context Information Management" factor group, because of their complementing relationship.

This grouping made easier the analysis process due to the great number of factors found during the execution of the systematic review.

The analysis of the papers returned by the second systematic review was made up in three steps:

- Identifying the presence of the ubiquitous characteristic;
- Identifying the factors of each characteristic, and;
- Grouping corresponding factors in factors group.

The results of the first and second steps are presented in Table 1. The first column shows the characteristics obtained from the first systematic review. The second and third columns show how many papers are concerned with each characteristic, in absolute values and percentage (each characteristic could be found in more than one paper, thus, the total of percentage found for each characteristic can be more than 100%). The fourth and fifth columns show how many factors were found for each characteristic. Finally, the sixth column shows the percentage distribution of factors per characteristic.

Table 1. Ubiquitous characteristics presence and factors.

Ubiquitous Characteristic	Presence	% of 59	Functional	Restrictive	% of 168
Service omnipresence	28	47.5	9	1	6.0
Invisibility	26	44.0	8	2	6.0
Context sensitivity	56	94.9	22	8	17.9
Adaptable behavior	52	88.1	24	8	19.0
Experience capture	11	18.6	7	0	4.2
Service discovery	28	47.5	13	13	15.5
Function composition	19	32.2	18	5	13.7
Spontaneous interoperability	21	35.6	10	2	7.1
Heterogeneity of devices	18	30.5	9	3	7.1
Fault tolerance	11	18.6	3	3	3.6
Total of factors			**123**	**45**	**100%**

According to the third step of the analyses, an example of factors group for the context sensitive characteristic is presented below:

Characteristic: Context sensitive
Factor Group: Information capture
Factor: To capture the user identity, location, effect or activity.
Factor: To consider the time variable.
Factor Group: Context information management
Factor: To contextualize the obtained information.
Factor: To store the captured information.
Factor: To consider semantics in the organization and capturing of the context information.
Factor Group: Sharing of information
Factor: To share context information with users and other devices.

The third and sixth columns of Table 1 allowed extracting the data to produce the graph on Figure 1. This graph represents the presence of each characteristic regarding the selected papers (third column) and the distribution of factors per characteristic (sixth column). It is possible to observe that the curves present similar behaviors. These behaviors, complemented by the results from the systematic literature review, led us to suggest that [Spínola et al. 2006]: (1) ubiquitous characteristics make sense, since none of them was discarded when classifying the 59 papers; (2) factors distribution is fair, since the most explored characteristics are that with the great number of factors; (3) relative importance of each characteristic in the current ubiquitous computing scenario does exist, since there are a significant difference in the presence of each characteristic.

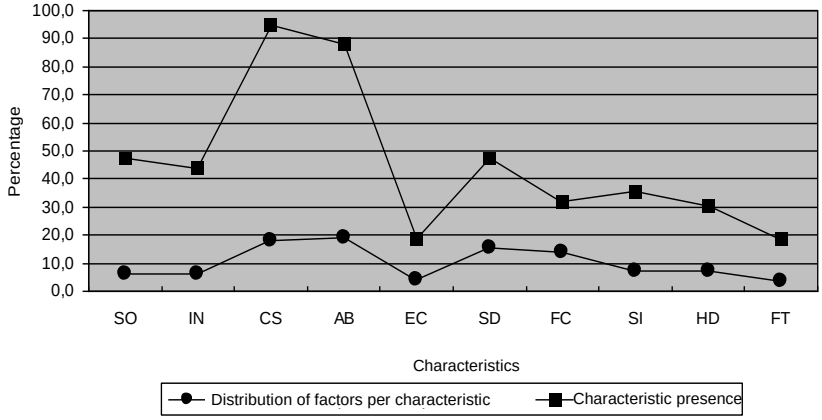


Figure 1. Distribution of Factors and Characteristics presence [Spínola et al. 2006].

5. An Approach to Characterize Software Projects According to Ubiquity

Based on the different perspectives presented on the papers selected by the systematic reviews described on sections 3 and 4, it was noticed that ubiquitous computing comes usually up in its totality when its ten characteristics are fully implemented in software projects. Thus, in a first analysis and according to our perspective, to be considered totally ubiquitous, a software project should contemplate the different factors of each ubiquitous characteristic. However, we can also have ubiquitous software projects with different levels of adherence to the ubiquitous characteristics. These different levels of adherence can be a consequence of the application domain and project's requirements, for instance. It is also important to say that those different levels of adherence could have as consequence the absence of some ubiquitous characteristic on the software project. This way, it is possible to have software project with different levels of ubiquity. However, it is important to notice that this paper does not intend to define whether a software project is more ubiquitous than other. Its goal concerns on observing how the different ubiquitous computing characteristics have been captured in software projects, supporting some understanding on how they could influence the software project planning.

Taking into account the concepts described in section 4, it was designed a checklist to characterize software projects according to their ubiquitous adherence level. This characterization comprises three steps: (1) to check the presence of the functional and restrictive factors of each characteristic; (2) to consolidate the software project adherence level of each characteristic based on the presence/absence of each functional and restrictive factor, and; (3) to generate the graph that will represent the application adherence level considering the ubiquitous characteristics (using the values calculated on step 2). To support the characterization steps, it has been built a spreadsheet-based form to calculate the adherence level for each characteristic. Doing so, the steps 2 and 3 are dealt trough an automated tool represented by a spreadsheet-based software.

Table 3 shows a fragment of the form that allows software engineers to capture the following information:

- Ubiquitous computing characteristic: shows the characteristics identified in the first systematic review presented on section 3;

- Adherence Level: shows the percentage of adherence based on the Status column. Notice that in this initial proposal each factor has the same weight and the adherence level is calculated as the average of the attended factors. The calculus is given by the expression bellow:

$$\text{Adherence Level} = \frac{\sum \text{attended factors} \times 100}{\text{Number of factors}}$$

  Where:
  - Attended factors are the factors whose status value is 1 for a specific characteristic;
  - Number of factors is the total number of identified factors for a specific characteristic.
- Factors group: shows the factors groups identified in the second systematic review presented on section 4;
- Factor: shows the functional and restrictive factors identified in the second systematic review presented on section 5;
- Status: factor presence (1) or absence (0). The software engineer provides this information.

A complete version of the checklist can be found at http://www.cos.ufrj.br/~ros/ubforms.html.

As the user fill out the Status column, the Adherence Level column can be calculated for each ubiquitous computing characteristic. As a final step, the evaluated percentage values are used to draw a graph that represents the adherence level of the software project according to the perspective of ubiquity. For instance, Figure 2 represents the obtained result when applying this checklist to the self-nominated ubiquitous application presented in [Kindberg et al. 2000].

Table 3. A checklist fragment to characterize ubiquitous software projects.

Characteristic	Adherence Level	Factor group	Factor	Status
SO	%	Mobility	User section management	1 or 0
			To deal with the user's mobility	1 or 0
			Factor n	1 or 0
		Service management	…	…
		Group n	…	…
IN	%	…	…	1 or 0
CS	%	Information capture	To capture the user identity, location, effect or activity	
			To consider the time variable	
			Factor n	
		Context information management	…	
		Group n	…	1 or 0
…	%	…	…	1 or 0

In the next section will be discussed the results of applying this checklist to classify the 8 projects found by the first systematic review (section 3) and that have been

not used to identify the functional and restrictive factors and did not influence either contribute for second systematic review's results.

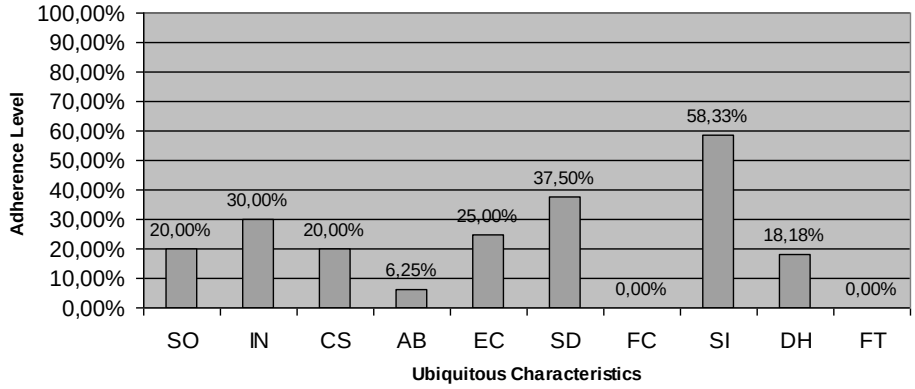
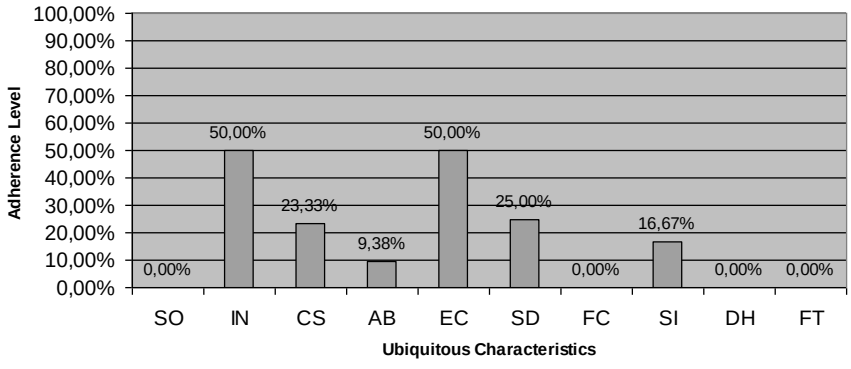


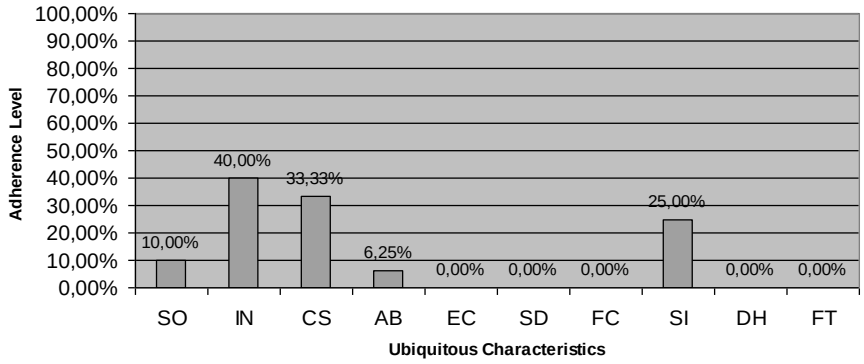
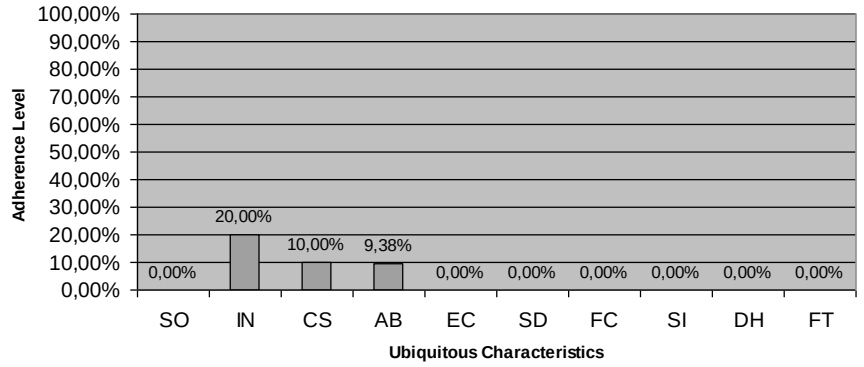
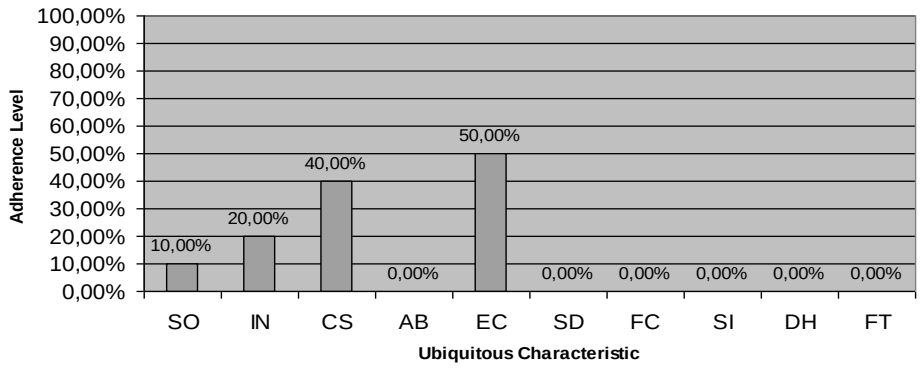
Figure 2. Example of characteristics and their adherence levels.

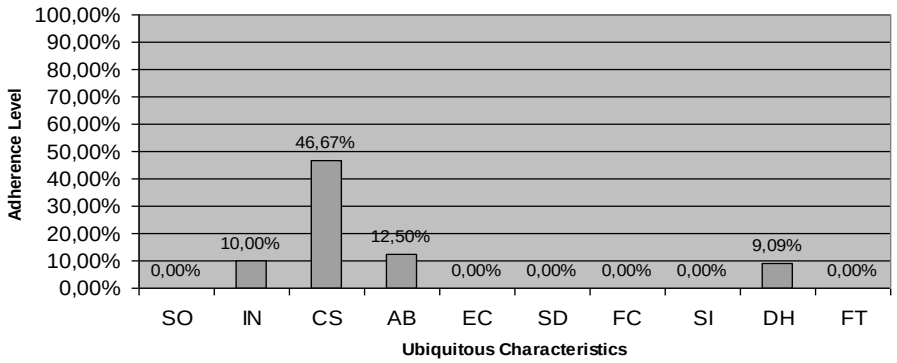
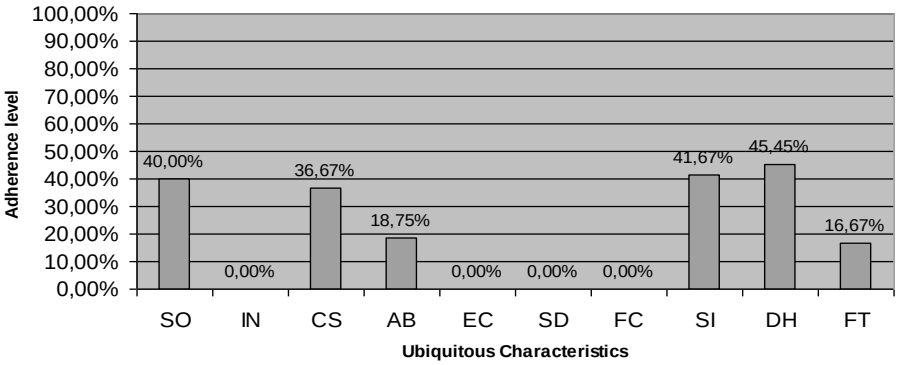
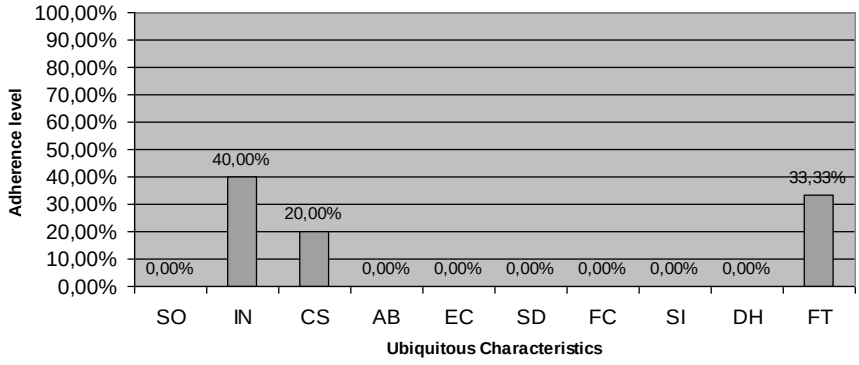
6. Applying the Checklist

In this section, the proposed checklist has been applied on a set of eight applications [Tahti et al. 2004] [Kindberg et al. 2000] [Ali et al. 2004] [Bossen and Jorgensen 2004] [Hatala et al. 2005] [Lee and Chung 2004] [Zhou et al. 2005] [Joel et al. 2004] characterized as ubiquitous in the first systematic review. These applications were found only in the first systematic review. For each one of the software projects, a characterization like that exemplified in Figure 2 has been made. In Table 4, it is presented the description and the ubiquitous adherence level of each software project analyzed.

Table 4. Description of the analyzed software projects.

Software Project 1 [Tahti et al. 2004]	
Description	This software project intends to develop a context sensitive application that supports the execution of office-type activities (use of printers, for instance). The application consists of a "personal assistant" composed by basic functions and several context sensitive services.
Ubiquitous adherence level	
Software Project 2 [Kindberg et al. 2000]	
Description	The Cooltown project intends to integrate web services with the purpose of improving the communication of the people, supplying located services according to the environment surrounding them and providing interaction with the objects found in these environments. The developed software allows the visitors of a museum to access the information using a PDA. Sensors are used to accomplish the communication between museum's objects and PDA.

Ubiquitous adherence level	See Figure 2.
Software Project 3 [Ali et al. 2004]	
Description	This software project presents a scenario where several services in the context of a kitchen could be made available using concepts of ubiquitous computation.
Ubiquitous adherence level	
Software Project 4 [Bossen and Jorgensen 2004]	
Description	This project goal is to construct software that allows an easy access to mobile electronic registration of patients (EPR).
Ubiquitous adherence level	
Software Project 5 [Hatala et al. 2005]	
Description	The goal of this software project is to develop an application to support user interaction in museums. The idea is to create a semantic net to integrate the different types of media (video, sound, and image) through the definition of it semantics and, turn them available to the users as they visit the different objects and areas of the museum.
Ubiquitous adherence level	

Software Project 6 [Lee and Chung 2004]	
Description	This software project intends to develop a set of applications regarding the concepts of the ubiquitous computing in the context of healthcare, environment control, management, and digital pictures.
Ubiquitous adherence level	
Software Project 7 [Zhou et al. 2005]	
Description	EZCab is a ubiquitous application that allows people make calls to taxis that are close and available, using cellular or PDAs. The system discovers and makes the call to available taxis.
Ubiquitous adherence level	
Software Project 8 [Joel et al. 2004]	
Description	This software project intends to develop software to support distance-based education.
Ubiquitous adherence level	

The data used to reach the results of Table 4 are summarized on Table 5. From Table 5 it was generated the graph on Figure 3. This graph represents the number of ubiquitous attended factors of each investigated application. It also shows the total number of identified factors for each characteristic (dashed horizontal lines).

Observing the graph on Figure 3, it is possible to notice an expected behavior: if the number of factors identified for each characteristic increases or decreases (dashed horizontal lines), the same happens with the number of factors implemented in the applications (vertical bars). The only exception to this behavior is the function composition characteristic. This behavior was not expected by the fact that this characteristic has been considered necessary in about 32.2 % of the analyzed papers of the second systematic review. However, a possible explanation could be the difficulty to deal with the inherent function composition characteristic factors complexity. Additional observation is regarding the concern on software projects. It seems that ubiquitous software projects pay more attention to the invisibility, context sensitive and adaptable behavior characteristics. The other characteristics seem to appear as isolated initiatives yet. As a final observation, we can say that this behavior resembles that one described at the end of Section 4 (concerned with the behaviors observed on Figure 1).

Table 5. Ubiquitous factors per application.

Ubiquitous Characteristic	Number of Identified Factors	Attended Factors per Application							
		SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8
SO	10	0	3	2	0	6	0	3	0
IN	10	4	3	4	3	3	2	0	4
CS	30	5	10	12	10	10	15	13	13
AB	32	3	2	2	3	0	8	10	0
EC	7	4	2	0	0	4	0	0	0
SD	26	9	12	0	0	0	0	0	0
FC	23	0	0	0	0	0	0	0	0
SI	12	3	6	3	0	0	0	4	0
HD	12	0	4	0	0	0	1	5	0
FT	6	0	0	0	0	0	0	2	1

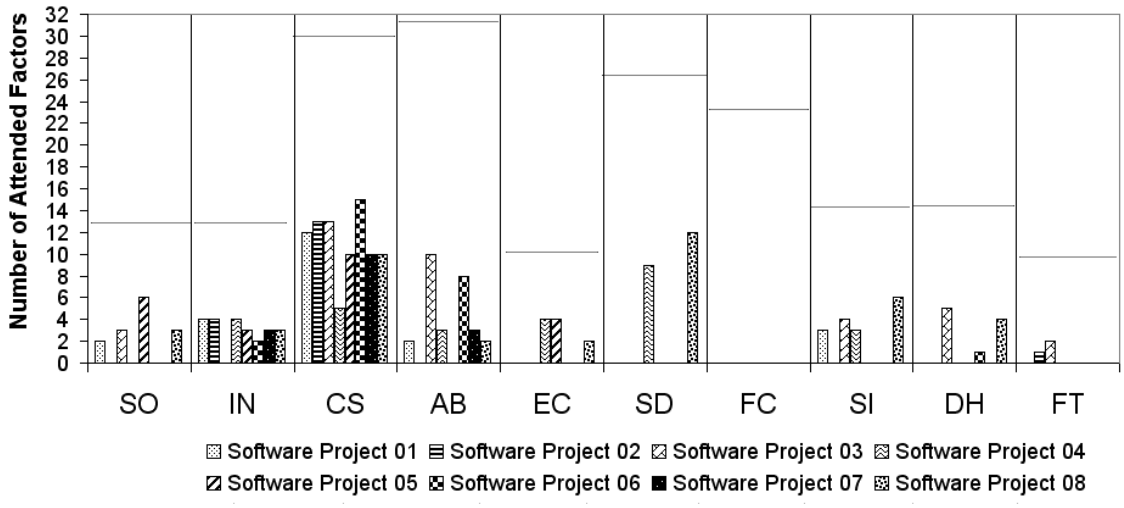


Figure 3. Attended factors.

7. Final Comments

The extent and the level of abstraction of the principles that guided the initial work of [Weiser 1991] allow many interpretations of what is ubiquitous computing. It can be clearly identified from the researches on ubiquitous computing selected by the

systematic reviews and presented in this paper. This way, the research aiming at the design of this checklist brings together, besides the checklist, some contributions:

- A current definition for ubiquitous computing and ubiquitous systems;
- A proposal of a set of concrete characteristics to achieve ubiquitous computing;
- Identification of functional and restrictive factors for each ubiquitous characteristic;
- A proposal of a checklist of ubiquitous software projects using the ubiquitous characteristic as a way to measure the adherence level.

Besides that, the first systematic review allowed us to conclude that ubiquitous software projects are too much restricted on their scope; they are still limited to research centers, and; they present solutions that take into consideration just a small number of ubiquitous characteristics and its respective factors.

The results obtained with the execution of the checklist are particularly important because they allow us to create a baseline that reflects the distance between the concepts effectively implemented on ubiquitous applications and the characteristics that define an application as ubiquitous. Certainly, the application domain determines that not all the ubiquitous characteristics shall be or need to be implemented. However, the low adherence level can provide indications about the difficulties found in the attempt to deal with these characteristics' factors. These difficulties can be directly related to the inherent complexity of each one of these characteristics and that has direct consequences in the project of the application, in the adopted software architecture, in the final product quality control, in the management of the involved resources, in the technologies adopted in the solution and, in the software development process. Into this context, several questions arise:

(1) Which are the dependencies and complementarities among functional and restrictive factors of the different characteristics?
(2) What are the essential characteristics of ubiquitous systems?
(3) Shall the factors groups be evaluated in the same level or the use of different weights for each factors group can improve the adherence level evaluation?
(4) Which are the influences of application domains on the relative importance of each ubiquitous characteristic? Which weights can be used according to the different application domains?
(5) Which are the impacts of each ubiquitous characteristic and its factors on the software architecture and design?
(6) Which (new) software engineering areas can provide support to the construction of ubiquitous software projects?
(7) How do we test and make sure that such software fulfills its specification?
(8) How can we increase the reliability of such ubiquitous environment?
(9) How do we track down and debug the cause of fails than have it fixed?

As stated by Sakamura (2006), the problem with the creation of ubiquitous software applications is very severe. Thus, the identification of application characteristics is important because they provide subsidies to the software planning and development phases that take into consideration the particularities associated with ubiquitous systems. It is important to enforce that the characteristics and the classification approach represent just starting points to new research activities regarding ubiquitous computing.

Acknowledgments

The authors would like to thank CAPES and CNPq (Experimental Software Engineering Grant: 472135/2004-0) for the financial support to this work. This work has been partially developed in collaboration with HP Brazil R&D.

References

Ali, J.A., Won-Sik, Y., Jai-Hoon, K., We-Duke, C. (2004) "U-kitchen: application scenario". Proceedings of the Second IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems, pp.: 169 – 171.

Biolchini, J., Mian, P.G., Natali, A.C.C., Travassos, G.H. (2005) "Systematic Review in Software Engineering". Technical Report ES 679/05. COPPE/UFRJ.

Bossen, C., Jorgensen, J.B. (2004) "Context-descriptive prototypes and their application to medicine administration". Proceedings of the 2004 conference on Designing interactive systems: processes, practices, methods, and techniques. Pages: 297 – 306.

Hatala, M., Wakkary, R., Kalantari, L. (2005) "Rules and Ontologies in Support of Real-time Ubiquitous Application". Journal of Web Semantics.

Joel, S., Arnott, J.L., Hine, N.A., Ingvarsson, H., Rentoul, R., Schofield, S. (2004). "A framework for analyzing interactivity in a remote access field exploration system". SMC(3) 2004: 2669-2674.

Kindberg, T., Barton, J., Becker, G., Caswell, D., Debaty, P., Gopal, G., Frig, M., Krishnan, V., Morris, H., Schettino, J., Serra, B., Spasojevic, M. (2000) "People, places, things: Web presence for the real world". Third IEEE Workshop on Mobile Computing Systems and Applications, pp.: 19 – 28.

Kitchenham, B. (2004) "Procedures for Performing Systematic Reviews". Technical report, Keele University, Australia.

Lee, S.H., Chung, T.C. (2004) "System Architecture for Context-Aware Home Application". WSTFEUS 2004.

Sakamura, K. (2006) "Challenges in the Age of Ubiquitous Computing: A Case Study of T-Engine, An Open Development Platform for Embedded Systems". Proceeding of the 28th International Conference on Software Engineering (ICSE), Shanghai, China. Pages: 713 - 720.

Spínola, R.O., Silva, J.L.M., Travassos, G.H. (2006) "Towards a Conceptual Framework to Classify Ubiquitous Software Projects". Proceedings of the 8th International Conference on Software Engineering and Knowledge Engineering (SEKE), San Francisco, USA.

Tahti, M., Rauto, V., Arhippainen, L. (2004) "Utilizing context-awareness in office-type working life". Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia.

Weiser M. (1991) "The Computer for the 21st Century". Scientific American, pp. 94-104.

Zhou, P., Nadeem, T., Kang, P., Borcea, C., Iftode, L. (2005) "EZCab: A Cab Booking Application Using Short-Range Wireless Communication". PerCom: 27-38.