EAI Endorsed Transactions
on Context-aware Systems and Applications
Research Article EAI.EU

Quality of Context in Context-Aware Systems

Asma A.Q. Al-Shargabi[1], Francois Siewe[2,*] and Ammar T. Zahary[3]

[1]Faculty of Computing and Information Technology, University of Science and Technology, Sana'a, Republic of Yemen; a.alshargabi@gmail.com
[2]Faculty of Technology, De Montfort University, Leicester, UK; fsiewe@dmu.ac.uk
[3]Faculty of Computer and Information Technology, Sana'a University, Sana'a, Republic of Yemen; aalzahary@gmail.com

Abstract

Context-aware Systems (CASs) are becoming increasingly popular and can be found in the areas of wearable computing, mobile computing, robotics, adaptive and intelligent user interfaces. Sensors are the corner stone of context capturing however, sensed context data are commonly prone to imperfection due to the technical limitations of sensors, their availability, dysfunction, and highly dynamic nature of environment. Consequently, sensed context data might be imprecise, erroneous, conflicting, or simply missing. To limit the impact of context imperfection on the behavior of a context-aware system, a notion of Quality of Context (QoC) is used to measure quality of any information that is used as context information. Adaptation is performed only if the context data used in the decision-making has an appropriate quality level. This paper reports an analytical review the state of the art on quality of context in context-aware systems and points to future research directions.

Keywords: Quality of Context (QoC); Context-Aware System (CAS); Quality Parameters; Quality Control, Ubiquitous Computing.

Received on 08 July 2016, accepted on 08 November 2016, published on 06 July 2017

Copyright © 2017 Asma A.Q. Al-Shargabi *et al.*, licensed to EAI. This is an open access article distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/3.0/), which permits unlimited use, distribution and reproduction in any medium so long as the original work is properly cited.

doi: 10.4108/eai.6-7-2017.152761

1. Introduction

The term 'Ubiquitous', which means appearing or existing everywhere, was combined with the term "Computing" to form the term "Ubiquitous Computing", which is used to describe ICT (Information and Communication Technology) systems that enable information and tasks to be available everywhere. Devices should vanish into the background to make the user and his tasks the central focus rather than computing devices and technical issues [1, 2]. From 80s at the last century up to current, Mark Wiser (the founder of ubiquitous computing) vision about ubiquitous computing has already achieved remarkable success benefiting from the technology advances especially within networking, sensors and mobile devices areas. This vision of ubiquitous computing aims at making our life better, easier and simpler in invisible manner. Devices should disappear in the background but working continuously with us anywhere, anytime with no annoying. In fact, ubiquity itself was not the objective, supporting this intelligent environment in our daily decisions smoothly is the main objective. These computing systems should know where they are, know the user, and realize what the user wants and what should be done to meet those wants. It should be adaptive and proactive. It should be context-aware.

Context-Aware Systems (CASs) are a field in the wide range of ubiquitous computing. They are systems that are continually aware of their situation (or context) in their physical, virtual (ICT) and user environment. CASs are able to adapt their operations to the current context without explicit user intervention and thus aim at increasing usability and

*Corresponding author. fsiewe@dmu.ac.uk

effectiveness by taking environmental context into account [1, 2]. Despite that, this vision is well understood for many years, context-aware systems still suffer from poor performances. They face many challenges due to many factors such as sensor shortages, rapid dynamic environment, lack of harmony between different sensors and difficulties faced in situation capturing. Many researchers investigated factors that can affect quality of context-aware systems from different views. A context-aware environment is complicated to some extent; there are different constructs that can affect the whole performance, in addition to interdependency between them [3]. Researches have addressed quality issues concentrating on variant dimensions ranging from quality of context to quality of service ending with quality of the devices which are used to acquire context. Quality of Experience (QoE) is addressed also recently as an aggregate of QoS [4, 5]. Quality of Measurements is also a hot topic that is addressed by many recent researches [6]. This paper addresses QoC.

Context-aware systems use context information to decide what adaptation actions to perform in response to changes in their environment. Depending on applications, context information includes physical context (e.g. temperature and location), user context (e.g. user preferences and user activity), and ICT context (e.g. device capabilities and battery power).

Sensors are the main means of capturing context. Unfortunately, sensed context data are commonly prone to imperfection due to the technical limitations of sensors, their availability, dysfunction, and the highly dynamic nature of environment. The roots of imperfection problem could be also formed by the diversity of context sensors that could lead to lack of harmony between different resources along with the technical shortages of sensors [7-9]. The openness of ubiquitous systems adds more challenges to context protection against many possible attacks [10]. Furthermore, sensors capture the context periodically, so some events could be easily missed between intervals assigned to sense the context. On the other hand, in the high level context (derived context), the reasoning rules that are used to derive context cannot be valid for all situations [11], thus the derived context could be invalid. This imperfection could lead to serious problems due to the wrong decisions that might be made accordingly [12].

Context imperfection aspects could also be addressed with profiled context that are created by the user; for example, the user could leave his/her agenda without updating for long time where it has actually changed [8]. Consequently, sensed context data might be imprecise, erroneous, conflicting, or simply missing [10, 13]. To limit the impact of context imperfection on the behavior of a context-aware system, a notion of Quality of Context (QoC) is used to measure quality of any information that is used as context information [3,14]. Adaptation is performed only if the context data used in the decision-making has an appropriate quality level.

This paper presents an analytical review of the state of the art on quality of context in context-aware systems. It covers many issues related to QoC in CASs including QoC definitions, quality parameters and their quantification methods, quality evaluation, quality policy, quality control processes, context quality management and quality assurance. The paper discusses current researches in QoC of CASs and points to future research directions.

The remaining of the paper is organized as follows. Section 2 gives an overview of ubiquitous computing. Section 3 reviews the research trends in ubiquitous computing. Context-aware systems are presented in Section 4, while Section 5 gives a comprehensive review of existing quality of context models. Section 6 discusses quality of context management, while Section 7 and Section 8 critically analyse the research trends in context-aware systems. Section 9 concludes the paper.

2. Ubiquitous Computing

The terminology ubiquitous computing (UC) is introduced for first time by Mark Weiser (1952 -1999). He worked at the Xerox Palo Alto Research Centre (PARC). PARC was the birthplace for many inventions that characterized the PC era such as the mouse; windows based interfaces and laser printers. Mark Weiser idea of UC first appeared in his famous article "The Computer of the 21st Century" published in Scientific American in 1991. The most frequently cited quotation from this article is the following paragraph: “The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it”.

The term ‘ubiquitous’, is a Latin word which mean "anytime" and "anywhere" [15]. It has been combined with computing to form the term "Ubiquitous Computing" which is used to describe ICT

(Information and Communication Technology) systems that enable information and tasks to be available everywhere, and to support intuitive human usage while appearing invisible to the user. Devices should vanish into the background to make user and his/her tasks the central focus rather than computing devices and technical issues [1, 2, 15-17]. In UC environment, computers are embedded in everyday objects to support daily activities at work, home, or anywhere. It creates a new environment from fusion between the physical world and the electronic space [15, 18-20]. Furthermore, in UC, information is not only served to user but also (proactive) actions can be taken based on the understanding of the current context situation.

UC suggests tiny, wirelessly intercommunicating microprocessors which are invisibly embedded into objects around us. Equipped with different types of sensors, these computers can record the environment where the objects are and provide them with intelligent processing capabilities and natural interactions [18,21]. Computer power and IT can be applied to all areas ranging from military and industrial production up to personal everyday life with a new quality [21].

IT innovations have gone through four main generations: mainframe, personal computer, distributed computing, and UC [19]. UC has seen a remarkable development where the physical world environment is being increasingly instrumented in a digital way and strewn with embedded sensor-based and control devices [1]. It is a natural result of recent advances in computers hardware and software technologies where many and variant devices with a wide range of computing, communication and storage capabilities have been invented [15]. UC is a typical crosscutting technology; it utilizes the whole range of modern information and communication technologies (ICT) such as microelectronics, the energy supply in user interfaces, information security, sensors and localization technology [21].

UC is the next wave of computing after the Internet wave. It aims at revolutionizing the current modes of human computer interaction. Computers have been already used in different aspects of human lives; however people have to adapt their behavior according to each computer system. In contrast, in UC, computing systems are invisible and embedded in each object around us in our daily life [22]. UC introduces a new paradigm of interaction. Authors in [23] addressed three types of interaction themes: natural interfaces, context-aware application, and automated context capturing and automated access.

UC is reflected in a many different concepts such as "pervasive computing", "ambient intelligence", and "the Internet of things". However, the common of all concepts is the goal of assisting people to make the life better and easier by using a numerous microprocessors and sensors integrated into the environment [21, 22].

Benefits of UC are introducing unobtrusive computing assistance to us when we navigate through our work and personal daily lives [24]; enabling us to retrieve information from anywhere in our real world that could not be available before and enabling us to control daily life objects surrounds us in a way that has not been done before. The main goal is reducing the complexity in our daily life [15].

One field in the wide range of UC is context-aware systems (CASs). CASs are able to adapt their operations to the current context without explicit user intervention and thus they aim at increasing usability and effectiveness by taking environmental context into account [2]. CASs will be introduced with more details in Section 3.

2.1 Main Aspects of UC

According to Mark Weiser, in UC, computers should be so imbedded, so fitting, so natural, that we use it without even thinking about it [25]. This leads to figure out the main aspects of UC [18, 25]. Nanotechnology and wireless technology is compound to create a seamless connection and invisible computing assistance, and context-awareness and natural computing to make the computers helpful and unobtrusive [25]. Aspects of UC are not completely distinct; they are strongly interrelated to each other. The following section introduces the main aspects of UC in details:

2.1.1 Ubiquitous Access/Wireless Connection

Access to services and information should be anywhere and anytime [21]. The access to information will be ubiquitous, over time access devices become available, divers, and smaller. Decentralization of systems and their comprehensive networking should be a vital aspect for UC [21]. Access channel should increasingly become wireless and widespread [26, 27]. Each device in UC needs to limit the range of its

wireless communication to enable valuable wireless bandwidth reuse. At 1990s, there were no short-range wireless standards, but right now, there are many: Bluetooth, IrDA, Zigbee, and WiFi. These technologies enable wide deployment for devices within local ad hoc communication as in UC vision [18].

2.1.2 Natural Interaction

Mark Weiser said UC should provide a natural interaction. He found the known interaction styles at those days did not make invisibility possible, they force the user to learn how the machine interacts. These traditional modes of interaction made the computation as a separate activity apart of our daily activities [24, 18, and 19]. The idea behind natural interaction is to provide computer services without forcing the user to think about how to use computer to get those services [25].

2.1.3 Calm technology/ Invisibility/ Embedded systems

Calm technology is another term that Mark Weiser used to describe UC. Authors in [22] considered invisibility as the most important aspect of UC. Opposite to PC applications that virtualize our world, UC aims to push the computers back into background to be a part of our life and no need to virtualize the reality [18]. Computer hardware and software are embedded in other equipment and objects of daily use [28].Any computer can be linked with a network, this link should be achieved smoothly without users intervention, and finally, appropriate services are provided on the network at the right time through human friendly interface [19,24].

2.1.4 Miniaturization/ Nanotechnology

In UC, ICT components are becoming smaller [27]. If computers are to be everywhere, invisible, and obtrusive, they should be as small as possible. Miniaturization of computer components to an atomic scale is called nanotechnology. Nanotechnology focuses on building transistors using highly miniaturized components using individual atoms or molecules. These types of computers will allow impressive levels of computing because of the huge number of transistors that could be located in tiny packages [25].

2.1.5 Context-awareness/ Autonomous processing

Computing elements can now have sensors to measure the physical world and actuators that initiate physical response; these systems are called context-aware systems. In this environment, UC can function automatically and autonomously [21, 26, and 27]. Automatic recognition and autonomous processing of repetitive tasks can be achieved without user intervention [21]. This means that computers should be able to accurately understand the user needs and provide him/her with the required services on time. The notion of context-awareness is that the computers will be able to understand the situation that they are located in to offer the proper services relevant to the current context. The attributes of context vary widely; beside the user, context may include location, user rules, current time, current date, and any other physical objects or peoples. The application of context-aware systems can be a coffee maker, a heating system at home or different GPS maps in cars or mobiles, or even earthquakes and flooding forecasting systems [25]. According to authors in [27], energy autarky and the autonomy of components and systems are considered as secondary characteristics of UC.

3. Trends of Research in UC

To realize the vision of Mark Weiser, UC still faces many challenges. Researches differed in their classification of the UC general trends. The following section summarizes these different trends. Of course, these research trends are not completely separated.

3.1 Wireless Problems

Until now, the different companies that produce the wireless technologies desire to produce their own proprietary products speaking their own proprietary language. This leads to “no interoperability” between devices from different companies [29]. Consequently, there are too many similar wireless standards. Within UC environment, we use many interconnected wireless technologies with high communication level. The challenge is “how to integrate these technologies that are based on different standards within UC environment?” [30].

3.2 Disappearing hardware / Mobile and Wearable Systems

UC systems are designed to operate opportunistically in the background of the user environment [31]. Technological advances will lead to the development of new hardware components that function more

invisibly than before [32]. Wearable technology is an important form of disappearing technologies. For example, research in wearable technologies introduced the concept of continually worn interfaces [31]. However, wearable equipment is still too clumsy and has limited field of vision, contrast and resolution. Wearable equipment needs being developed to be lighter, smaller and easier to work with [33].

3.3 Efficiency and Reliability

There are general tendencies from research community towards highlighting the quality attributes and performance of UC systems, especially efficiency and reliability. Authors in [30] considered the efficient soft computing techniques as most important UC research trend. Authors in [33] consider the challenges related to system properties as one important research trend. According to [33], these properties are such as response time delays, hardware or software failures. This attention is understandable as UC systems should act 24 hrs/day, they should possess high level of availability, reliability and efficiency.

Authors in [29] introduce a clear analogy for UC system's reliability. Today's personal computers are, in a sense, becoming more and more reliable. However, they still have a long road ahead for them to catch up with the reliability exhibited by other well-founded technologies, such as televisions, telephones and even washing machines. These well-founded technologies have been successful, in part, due to their reliability [29]. This analogy clearly indicates the level of reliability we want to realize for UC systems.

3.4 Context-aware Systems

The importance of adaptation to the context is understood in the field of mobile computing but, in UC environment, this is more complex where there is a need to respond to a much larger set of contextual triggers [34]. According to Weiser's different scenarios of UC environment, we can discern a form of intelligence where the system can predict the user's tasks and control and coordinates different actions to help the users. Making UC systems as context-aware is still unsolved problem [30, 34].

Beyond that, how accurate UC systems succeed in representing the real world is one important challenge. Occlusion detection is an active area of study of UC systems. Using computer vision in combination with sensors could provide promising results. However, it is mostly a top-down process and hard to deal with object dynamics, and evaluation of different hypotheses [33]. Although one can find an infrastructure with well-grounded technologies, this environment is not constructed buildings equipped with devices to support smart environment and pervasive computing [29].

3.5 Privacy and Security Issues

Without a doubt, the subject of the security and privacy is one of the most important topics for future of UC. It is expected that UC will have a social impact on our society just as the previous two eras of computing did. However, how will it affect privacy? Will society turn to a social solution, legal solution, ethical solution, or technological solution to protect privacy? [29].

The traditional approaches of privacy and security are inadequate for a modern, open information society. For example, to demand that sensitive data be deleted after its use is clearly out of sync with the Internet. Most important, legislation must acknowledge that person-related data have become a currency in the information economy. Here lies a core problem for developers who need to create systems that better address privacy issues. Currently, users don't fully understand how the electronic trails they create can be used, so they cannot understand their personal data's value. A key challenge for future ubiquitous system designers is to empower users to evaluate the trade-off between protection of privacy and access to improved services. Meanwhile, legislation must contribute by defining the boundaries within which such trade-offs may occur [34]. On the other hand, private information should be encrypted before transferring to the network administrator. Data security must be assigned higher priority because other issues may harm users but data leak may scare the user [35].

One more ambitious goal would be to provide users with a "sixth sense" that alerts them to serious privacy threats. In the real world, such mechanisms play a key role in survival. If we can provide a sixth sense for pervasive computing, we can avoid many serious threats. This sixth sense could be provided by studying the data patterns in the previous context history and benefiting these patterns to predict the potential threats. On the other hand we must revisit

many authentication and authorization mechanisms in the context of pervasive computing. For example, what authentication techniques are best suited to pervasive computing [36].

Another important security issue is the security policy. In UC security policy policies are context-dependent which makes it difficult to design enforcement mechanism that are simultaneously unobtrusive to the user while able to discover available services in a transparent manner [30].

3.6 Robotics/ Embedded Systems

Robotics is an emerging field that mobilizes a computer and enables it to effect change at arbitrary locations in the real world [30, 32]. This form of mobility and computer aiding is one important feature for UC. Robotics and embedded systems are the practical form of this feature.

3.7 Computer Interface

The current forms of computer interface are not suitable for UC in many ways. Until now, human activities are performed within two separated spaces, the physical world and the cyber world. Although many activities can be done faster and more accurately using current technology, many other activities are still manually done. Moreover, the systems in UC environment continuously operate in the background; this type of systems should be designed with continuously present computer interface [31]. Novel natural and continuous interaction modalities such as speech become a necessary component because they do not require bulky displays or input devices. There is much work left to be done to fuse physical and the cyber world together seamlessly and invent new natural and continuous modes of interfaces that facilitates the daily tasks without disturbing users [31, 32].

3.8 Evaluating UC systems

The evaluation of UC systems still faces some challenges. This is why there is little published work on UC evaluation. UC systems are complicated and the evaluation factors include a wide range criteria and different perspectives. Therefore, formative and summative evaluation is difficult. In this situation, focusing on the important factors would be necessary. Among the different views of evaluation, this research recommended a task–centric and use need approach for evaluation [31].

3.9 UC Management Mechanisms

As the number of deployed components increases, system management will likely become increasingly problematic. While we want zero-configuration, low-maintenance systems, the reality is that substantial system management will still likely be required. For many components, the administrative domain might change dynamically—for example, depending on the proximity of different users or devices. The combination of requirements for low (or zero) administration, multi domain management, and support for rapid reconfiguration will likely raise new challenges for system management [34].

On the other hand, most individuals who operate a personal computer have no knowledge of how to administer a single workstation. It would be unrealistic for the manufacturers of UC devices to expect their users to administer a complex network consisting of multiple devices. How does UC deal with this challenge? [29].

3.10 New Economic Models

One major challenge of UC is that none of UC scenarios seems to generate significant revenue. Consequently, UC systems will not receive the required financial support. Thinking about revenue in different way could help. The cost of deploying and operating a given component might need to be recovered in the form of many small contributions from applications that use the component. This could require support from components in terms of billing and accounting at a level previously unseen in widespread distributed systems [34].

3.11 Health Issues

IR technology is strongly used in UC systems for different purposes. Radiations problem will be one big health challenge in UC. This issue should be tackled seriously. We have to produce technologies that are environment friendly [30, 35].

4. Context-Aware Systems

In UC, to succeed in inventing helpful computing environment and natural interaction modes, we have to understand how humans interact with the environment. People are successful understanding each other using their rich language, their prior knowledge of how the world works and the implicit knowledge of the everyday situations and patterns. Currently, computers are not qualified enough to communicate with people and understand the context. When humans want to accomplish a task via computers, they should learn the method the computers understand to accomplish the task. We can say that they should "translate" what they want for each task. This is nothing in comparing with using human interaction styles. Improving the computer's ability to understand and acquiring context can help increasing the effectiveness of communication between computers and humans and help producing more helpful computational services.

4.1 Context-Awareness Concept

Many researches addressed context-awareness in two approaches: improving the human-computer interaction by enabling computers to interact in a much more natural manner, and enabling computers to understand the context situation of the environment that they are located in. Applications that capture and use context are said context-aware. The increasing availability of sensing technologies makes it easier to sense context for different environment situations. CASs are becoming more popular and can be found in the areas of wearable computing, mobile computing, robotics, adaptive and intelligent user interfaces, augmented reality, adaptive computing, intelligent environments, and context-sensitive interfaces [18].

Context awareness allows applications to be aware of the context/environment that they are located in, and react according to the best possible user experience [18]. It is the ability of computers to be perceptive, interpretive and reactive [16]. CASs are able to adapt their operations to the current context without explicit user intervention [1, 2]. In [18], authors defined context awareness with the following statement: "A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task" [18]. Therefore, CASs can capture context, view the captured context to the user or adapt according to the context situation and the application purposes.

According to authors in [18], CASs designing process can be summarized by this sequence of activities:

- Context Specification: determining behaviors that the application requires, in which situations each behavior should be executed and how;
- Context Acquisition: determining the hardware and/or software sensors that are required to acquire the context that is identified in the first step;
- Context Delivery: specifying how context should be delivered from the sensors to the applications that will use the context;
- Context Reception: based on the application purposes, specify what context is interested. This includes converting the context into a form usable by the application and then analyzing the context to determine whether this context, when combined with other available context, describes a relevant user situation for the application or not.

4.2 Context in Context-Aware Systems

There is a range of definitions for context. Authors in [37] define context as 'any information that can be used to characterize the situation of an entity that is considered relevant to the interaction between a user and an application'. According to authors in [22], context is the information that characterizes the current situation of environment for a participant in an interaction [22]. A more concreted definition of context is introduced by [38] as "a member from the set of context types, such as location, identities of nearby people, objects and changes to those objects".

Context information may be acquired in a variety of ways, such as applying sensors, network information, device status, user profiles and using other source [1]. This often depends on applications' use. Environment monitoring applications use multiple types of distributed sensors to determine an environment context such as air pollution and temperature [2]. Context is difficult to model because it contains many different dimensions such as location, time, located nearby devices, persons who are present, physical factors such as sound, motion, temperature … etc. [18]. Of course, this is not has to be the case for all applications, it is different from an application to another.

Several ways have been addressed in literatures to classify context. Authors in [39] and [24] classified

context into external and internal, which mean physical and user contexts respectively. Authors in [25] classified context into physical and logical contexts where the logical context is similar to the user context. Authors in [21] proposed three classes: (1) places such as rooms and buildings, (2) people, either individuals or groups, and (3) things such as physical objects and components. Authors in [16] classified context into three categories: (1) where you are (location context including which physical environment resources are located with the user), (2) who you are with (social context), and (3) what (ICT) resources are nearby. Authors in [86] introduces what they called "An Occupant-Centered Pragmatic Approach". This approach includes three aspects of context with some relations within each: (1) the first aspect is the physical environment around the occupant with the following relations: time, location, devices and people; (2) The activity of the occupant with mental and physical relations; (3) The physiological state of the occupant with preferences and feelings relations.

On the other hand, authors in [22] and [15] distinguished between static and dynamic context. Static context is invariant context such as a person's date of birth. Dynamic context can be highly variable over space and time, e.g. temperature. Context reasoning is also an active topic in context-aware systems [65] but it is out of scope of this paper.

5. Quality of Context (QoC)

This section introduces a review for different concepts and issued related to quality of context. These issues are the concepts of context quality, context imperfection, QoC parameters, QoC evaluation, and context management processes.

5.1 The Concept of Quality of Context (QoC)

Referencing to the quality issue in general, a specialist can use some verified heuristics to ensure conformity of required quality level when dealing with computing systems. These heuristics are always represented as data to facilitate computing quality realization and management in order to support the performance automatically.

As a result, quality of context is information about the context information that enables us to judge the quality level of context. In light of that, the first definition of QoC was introduced by [3] as: "Quality of Context (QoC) is any information that describes the quality of information that is used as context information". In our view, this definition is simple and cannot capture all and critical qualities aspects that affect and ensure quality of context. This definition focuses on the representation of quality more than the key quality aspects for context within a context-aware ubiquitous environment.

In our opinion, this definition of QoC guides researchers to elaborate the context quality as general and with concentrating on inherent objective information about context information apart of consumer view and the real context. In light of this definition, the researchers investigated context quality to cope with problems accompanied with imperfect context such as imprecision and erroneous. This approach seems not balanced and it does not satisfy the objectives of the context consumer. Thus, later, the definition of QoC is modified by [14] to involve the subjective nature to the concept with engaging user satisfaction to the definition: "Quality of context indicates the degree of conformity of the context collected by sensors to the prevailing situation in the environment and the requirements of a particular context consumer". This definition is better from the view that it highlights the sources and the consumers of the context and their needs and roles to ensure and realize quality. This view is closer to the reference model of quality and its nature. This led us to discuss how the definition should be coined, what are the factors involved. It is not a philosophy but the definition will affect all other quality issues ranging from parameters, indicators, measurement methods and even the quality policy.

5.2 Context Information Imperfection

Context in CASs has many sources of ambiguity. Sensors can sense incorrectly, fail, or be unsure about the context that they sensed. For higher context, context inference engines can inaccurately derive context situations or at least be unsure about their inferences. Other types of shortcomings are coming with profiled context which are created by user, up_to_dateness problem could be addressed [7-9, 11, 18].

Imperfection aspects could be: unknown (no available sensor information), ambiguous (conflicting information from different sources), imprecise

(information with insufficient granularity), or erroneous (sensed or aggregated context not coherent with real situation) [13, 40, 41]. Context imperfections may lead to problems in realizing both functional and non-functional properties in a CAS [42]. It could lead to serious problems due to the wrong decisions that might be made accordingly [12].

In our opinion, investigating reasons and aspects of imperfect context will contribute significantly when determining QoC parameters and quality level of context in general. The nature of the environment and the openness of CASs could be serious threats for the context integrity. Reasons related to context in high-level "derived context" received low attention by researchers despite it is considered more important for introducing the service.

5.3 QoC Parameters

QoC parameters are the attributes used to compute quality level of context. These parameters have been proposed to overcome the shortcomings of imperfect context such as ambiguity, imprecision, and up_to_dateness. The common parameters proposed by the literatures are reliability, up_to_datedness (timeliness), accuracy, completeness, security_level; significance, usability, representation_consistency, probability_of_correctness, and trustworthiness [3,14,7,8,12,43,44,45,46,47].

There are some parameters proposed in literatures with different titles but they define the same concept. It is important to address that to avoid repetition. We addressed three cases for that; (1) up_to_datedness and timeliness were used to reflect the freshness of the context for given purpose; (2) Precision, accuracy, and resolution are three titles define the granularity level of context; (3) Sensitiveness, access_rights, security_level are three titles for describing the security level as defined by the user.

Reliability is the probability of context being true according to sensor limitations [14, 44]. Timeliness indicates the degree of rationalism to use a context object for a specific application at a given time [3, 7, 14, and 44]. Accuracy is the level of details in which the context information characterizes the real world [3, 12, and 44]. Completeness indicates the quantity of information that is provided by a context object [7, 14, and 12]. Security_level indicates the extent to which owner of context allows the context consumer to access context [14, 12]. Significance indicates the worth or the preciousness of context information in a specific situation [14, 7]. Usability indicates suitability of use for an intended purpose [14]. Representation_consistency indicates the extent to which context representation format is consistent to consumer requirements [12]. Probability_of_correctness denotes the probability or the confidence that a piece of context information is correct based on its previous occurrences [3,47]. Trustworthiness also describes how likely it is that the provided information is correct; in comparison to the probability_of_correctness, however, trustworthiness is used by the context provider to rate the quality of the actor from which the context provider originally received the context information [3,7,46]. For derived context, authors in [42] proposed a definition for state reliability of a service as the probability that all components and connectors that implement it do not fail all the times they are used (i.e. numbers of activations over components and interactions over connectors). Table 1 shows an example explaining how the different literatures address QoC parameters. The level required from these quality parameters varies from an application to another based on the nature of context and CASs area.

Authors in [14] have introduced a special point of view for QoC parameters that distinguishes between objective and subjective parameters. The objective parameters encompass the quality requirements that are independent of the context consumer. They tell whether the collected context information is free of error and suitable to be used at an instance of time or not. On the contrary, subjective parameters are related to the user requirements and for a specific purpose. Objective parameters proposed in [14] include Reliability, Timeliness, and Completeness, while the proposed subjective parameters include Significance, Access_Rights, and Representation_Consistency. Classification of the parameters into objective and subjective will be useful when we want to assess the context validity, as it depends on the objective parameters.

When analyzing the changes in QoC parameters over time, some remarks can be raised. The first version of QoC parameters was coming with affecting of the first definition of context, which concentrates on its representation nature, as it is information, and also it was affected by the reasons focuses in sensors shortcoming to produce accurate data.

Therefore, all parameters except up_to_dateness were describing the accuracy of data. The later updates of these parameters were more engaging with the consumer requirement dimension and the measurement situation dimension to get more accurate view of the reality.

Table 1. Examples of QoC parameters in the literature

[3] 2003	**[44] 2006**	**[14] 2008**	**[7] 2008**	**[12] 2010**
Reliability				
-	Probability of information being true	The extent to which context can be considered credible	-	-
Accuracy				
-	Difference between the encoded and actual value of an attribute.	-	-	-
Timeliness/Up_to_datedness				
Up_to_dateness describes the age of context information	Acquisition Time and Validity Time provide the temporal reference of the information associated to the context item.	Indicates validity of context to use considering its freshness	Indicates the degree of rationalism to use a context object for a specific application at a given time.	-
Source				
-	Source of information, e.g. GPS	-	-	-
Precision				
Precision describes how exactly the provided context information mirrors the reality. Precision is specified with bounds. A GPS_receiver, for example, allows for a precision of about 4 meters.	The smallest measurable element.	-	-	The level of details in which the context information characterizes the real world.

5.4 QoC Evaluation

Many formulas for measuring QoC parameters have been proposed in the literature; mainly based on the quantification of these parameters [12, 14, 48-52]. Quantification means describing parameters with numeric values (decimal values within the range [0, 1] are usually adopted). This quantification is necessary for the following reasons: (1) for user, it is easier to deal with decimal values because it is more expressive and also it offers a useful scale for accurate measuring; (2) QoC measuring can be exploited for automatic context processing.

Authors in [14] use some information from sensor characteristics and specification and context consumers for QoC quantification. This information looks like the illustrated by Table 2. As explained above in this section, a metric can be objective or subjective. Table 3 summarizes QoC metrics and the calculation method.

Table 2. Sensor characteristics, specification and consumer requirements [14].

Sensor Characteristics	
Accuracy	Extent to which data is correct and free of errors
Precision	Degree of exactness with which context is collected
Granularity	Degree of detail with which context is collected
Time Period	Time interval between two readings of context
Sensor State	Physical state of sensor (static or dynamic)
Sensor Range / Span	Maximum distance for which sensor can collect context
Measurement Context	
Measurement Time	Time of collection of context information
Sensor Location	Location of sensor when context information is collected
Information Entity Location	Location of the real world entity about which context is collected at the time of collection of context
Available Attributes	Number of attributes that have a value for that context object
Specifications and Consumer Requirements	
Validity Time	Maximum length of time for which a specific type of context information is stable
Required Attributes	Number of attributes that are required to have a value for that type of context information
Critical Value	Level of importance of context information of a specific type
Access Level	Information about the rights of context consumer to access certain type of information

Table 3. A summary of QoC metrics and calculation methods [14].

QoC Metrics	Objctive view	Subjective View	Calculation Method
Reliability	X	-	Combination of span reliability and accuracy
Timeliness	X	X	Ratio of age and time period or validity time of context depending upon subjective or objective view
Completeness object	X	X	Ratio of available number of attributes to total or required number of attributes of a context depending upon subjective or objective view
Significance	-	X	Ratio of critical level of context to maximum critical level that type of context can have
Usability	-	X	Comparison of level granularity of context with level of granularity required by context consume
Access Right	-	X	Comparison of access level of context allowed by context owner to access level of context consumer
Representation Consistency	-	X	Comparison of representation formats

However, researches started introducing different sources and consequently formulas for some QoC parameters as these metrics and calculating methods are still not standardized and need a lot of effort by the research community to make them standardized. Authors in [12] added modifications to some formulas used for QoC parameters measuring which were introduced by [14]. For instance, according to [14] completeness parameter for a context object is calculated as a ratio between sum of weights of available attributes of a context object, and sum of weights of all attributes of that context object. In contrast, authors in [12] found this way more expensive and not realistic, as it needs to be recalculated again at each new reading of a sensor. Furthermore, this approach does not indicate if the context information is available and current. Therefore, authors in [12] proposed a measurement method for completeness that describes how the context information is complete, available, and current.

Table 4 summarizes the formulas introduced in the literatures to calculate QoC parameters (Formula 1 to Formula 14). All these formulas assumed the piece of context as context object and denoted to it as O.

Table 4. Calculation of QoC parameters.

Calculation Method for QoC Parameters	Ref
Reliability	
$$\boldsymbol{Reliability}\ (\boldsymbol{O}) = \begin{cases} \mathbf{1} - \frac{\boldsymbol{d}(\boldsymbol{s}-\boldsymbol{\varepsilon})}{\boldsymbol{d_{max}}} * \boldsymbol{\delta} & : \ \boldsymbol{if} \quad \boldsymbol{d}(\boldsymbol{s}-\boldsymbol{\varepsilon}) < \boldsymbol{d_{max}} \\ \mathbf{0} & : \ \boldsymbol{Otherwise} \end{cases}$$ (1) $\boldsymbol{where}\ \boldsymbol{d}(\boldsymbol{s}-\boldsymbol{\varepsilon})$ is the distance between context object and sensor, $\boldsymbol{d_{max}}$ is the maximum distance	[14]

and $\boldsymbol{\delta}$ is accuracy of sensor.	
Timeliness	
$$\boldsymbol{Timeliness\ (O)} = \begin{cases} \boldsymbol{1 - \frac{Age(O)}{ValidityTime(O)}} & \boldsymbol{:\ if\quad Age(O) < ValidityTime(O)} \\ \boldsymbol{0} & \boldsymbol{:\ Otherwise} \end{cases} \quad (2)$$ $$\boldsymbol{Age\ (O)} = \boldsymbol{T_{curr} - T_{meas}(O)} \quad (3)$$ Where $\boldsymbol{T_{curr}}$ is current time, and $\boldsymbol{T_{meas}(O)}$ is the time that the context object $\boldsymbol{O}$ was measured at it, and the context consumer determines *validity_time*.	. [14]
Completeness	
$$\boldsymbol{Completeness(O)} = \frac{\sum_{j=0}^{m} \boldsymbol{w_j(O)}}{\sum_{i=0}^{n} \boldsymbol{w_i(O)}} \quad (4)$$ Where *m* is the number of available attributes and *n* is the total number of attributes for context object $\boldsymbol{O}$. $$\boldsymbol{Completeness(O)} = \begin{cases} \boldsymbol{CO(O) \times \frac{NumberOfAnsweredRequest+1}{NumberOfRequest+1}} & \boldsymbol{:\ if\ U(O) \neq 0\ and\ O \neq null} \\ \boldsymbol{CO(O) \times \frac{NumberOfAnsweredRequest}{NumberOfRequest+1}} & \boldsymbol{:\ Otherwise} \end{cases} \quad (5)$$ Where *CO(O)* is completeness of a context object and *U(O)* is timeliness of object . The values *NumberOfAnsweredRequest* and *NumberOfRequest* can be obtained from a log file.	[14] [12]
Representation_Consistency	
$$\boldsymbol{RepresentationConsistency(O)} = \frac{\boldsymbol{k}}{\boldsymbol{TransformationEffort}} \quad (6)$$ Where $\boldsymbol{k}$ is a constant of normalization ranging in [0,1] depending on maximum and minimum values of transformation effort.	[14]
Resolution	
$$\boldsymbol{Resolution(O)} = \frac{\boldsymbol{current_granularity_level}}{\boldsymbol{total_number_of-granularity_level}} \quad (7)$$	[12]
Usability	
$$\boldsymbol{Usability(O)} = \begin{cases} \boldsymbol{1} & \boldsymbol{:\ if\quad GranularityLevel(O) \geq GranularityLevel(CR)} \\ \boldsymbol{0} & \boldsymbol{:\ Otherwise} \end{cases} \quad (8)$$ Where *GranularityLevel(CR)* is current granularity level and *GranularityLevel(O)* is granularity level of context object *O*.	[14]
Access_Rights	
$$\boldsymbol{AccessRight(O)} = \begin{cases} \boldsymbol{1} & \boldsymbol{:\ if\quad AccessLevel(O) \geq AccessLevel(CR)} \\ \boldsymbol{0} & \boldsymbol{:\ Otherwise} \end{cases} \quad (9)$$ $$\boldsymbol{AccessRight(O)} = \frac{\boldsymbol{current_security_level}}{\boldsymbol{total_number_of_security_level}} \quad (10)$$	[14] [12]
Probability_of_Correctness	
$$\boldsymbol{Probability_Of_Correctness(O)} = \sum_{i=1}^{m} \boldsymbol{confidence(Oi \Rightarrow O)} \quad (11)$$	[50]

$\boldsymbol{confidence(Oi \Rightarrow O)} = \frac{\boldsymbol{sigma(Oi \cap O)}}{\boldsymbol{sigma(Oi)}}$ (12) Where: $\boldsymbol{sigma(Oi \cap O)}$ is frequency of *Oi* and *O* occurring together in the previous history and $\boldsymbol{sigma(Oi)}$ is frequency of *Oi* in the previous history. *Oi* could be a single object and a combination of objects as *m* will include all possible combination of objects, which affirm $\boldsymbol{O}$ according to the previous history. $\boldsymbol{PoC(ce_i)} = \frac{1}{n+m}\left(\sum_{j=1}^{n} \boldsymbol{p(ce_i/ce_j)} + \sum_{k=1}^{m} \boldsymbol{QoCp(ce_{i)_k}}\right)$ (13) Where *ce* is the abbreviation for context element. *p(cei|cej)* is the conditional probability between the context element under investigation *cei* and the context element depends on *cej*. *DCi* is the totality of dependent context elements *cej*, which a context element under investigation *cei* depends on. QoCp(cei) is the QoC parameter of the context element under investigation *cei*. *n* is number context elements cei depends on (*n* = *DCi* size) and *m* is the number of QoC parameters.	[53]
Significance	
$\boldsymbol{Significance(O)} = \frac{\boldsymbol{CV(O)}}{\boldsymbol{CV_{max}(O)}}$ (14) Where *CV(O)* is the critical value of context object $\boldsymbol{O}$ and $\boldsymbol{CV_{max}(O)}$ is the maximum critical value that can be assigned (by the consumer) to context object *O*.	[14]

On the other hand, authors in [54] introduced the idea of context layers and context relative weights to compute quality parameters. Moving up the context layers, context facts are derived from source data. This idea led to compute some quality parameters value according to the position of piece of context in the context pyramid.

For example, each fact has a measurable context confidence that is derived from the underlying source data uncertainties. Situations are created by combining context facts. Situation confidence is calculated from the underlying context facts by combining the confidence of context facts with appropriate context weights, resulting in a single measure of situation confidence.

The purpose of context weight is to quantify a context fact's contribution to the occurrence of a situation, with respect to the other context facts of the situation. Authors in [55] introduced the Bayes' Theorem to calculate confidence. Then, they improved their work by introducing more fine-grain layers in [56] with three layers: sensor, abstracted context, and situation. The situation can be inferred from many intermediate situations, which are all together called abstracted context.

5.5 Calculating the Overall QoC

Subsection 5.3 illustrates different methods that are proposed to measure a QoC parameter for context object individually. However, many methods have been addressed in the literatures to calculate the overall quality of context. Authors in [12] proposed three basic methods: average value of QoC parameters, maximum, and minimum values depending on the nature of QoC parameters.

Authors in [48] introduced a fine-grain approach that evaluates quality of context. They focused on quality management and how to evaluate context. They proposed weighting idea for different quality attributes according to application type. The roots of weighting solution were introduced in [57] for Byzantine Generals Problem. Authors in [48] introduced an idea for weights scale by proposing a wide textual/numerical scale (7 levels) that describes the relative importance for QoC attribute compared to other quality attributes (Table 5).

Table 5. Textual/Numerical scale for relative importance of QoC attributes [48]

Relation	Value added to relative weight
equivalent	+0
Barely	+0.1
just a little	+0.2
somewhat	+0.5
Highly	+1
Much	+5
tremendously	+10

Then, it assigns the same weight for all QoC attributes and then adds the weights produced by comparing quality attributes to each other. It uses the final weights when computing quality of attributes for each context. To formulate that, let qi,j be the ith quality parameter for the jth context information, with $1 \le i \le n$ and $1 \le j \le p$. The overall QoC of the jth context information is illustrated by Formula 15 [48]:

$$QoCj = \frac{\sum_{i=1}^{n} wi * qij}{\sum_{i=1}^{n} wi} \qquad (15)$$

5.6 Quality Policy for Context in CASs

Policies are used widely in the literature for determining required levels of quality. For quality of context, two common types of policies are used: quality policies and procedural policies. However, the nature of policy is still the same; authors in [58] proposed policies which guide the behavior of entities within the policy domain. Quality policy is a set of quality rules that achieve quality objectives [58-61].

Authors in [62] introduced an approach for designing QoC policy. They built their approach based on the assumption that each CAS has its own objectives and a group of possible scenarios. Therefore, quality policy should not be confined of quality parameters alone; it should consider the application objectives beside quality parameters. Accordingly, two classes of quality policy are defined, application driven policy, and QoC policy.

Authors in [62] distinguished between two types of QoC policies, static policy and dynamic policy. Static policy should be used when the value of a quality parameter can be predefined whereas dynamic policy defines the value of a quality parameter during run-time.

Table 6. Examples of recommended QoC attributes for different types of context [48].

Context Category	Context Level	Quality Parameters
Social Activity Emergency	H	freshness accuracy
Space/Temporal	L, I, H	Accuracy precision
Identity Privacy	L, I, H	Trust_worthiness

Authors in [40] and [63] introduced a policy based approach for QoC. The idea behind this approach is to focus on determining the most important QoC attributes/indicators based on the nature of application and context type/class. It selects the context items (sensed raw data) among different sensors and then determines the context situation (derived context) according to these policies. Authors in [48] recommended some QoC attributes for different types of context/applications as illustrated in Table 6.

6. QoC Management

CASs should be able to adapt according to the context information captured from distributed redundant context sensors. For correct adaption, these systems should ensure quality level of context information that characterizes different situations. This subsection is devoted to demonstrate the work that is introduced by research community for context management and context quality assurance and control.

6.1 QoC Assurance and QoC Control

According to ISO 9000 definitions [64], Quality Assurance (QA) is "a part of quality management focused on providing confidence that quality requirements will be fulfilled". Quality control is defined as: "A part of quality management focused on fulfilling quality requirements" [64,5]. Elements such as managed processes, criteria, and qualifications competence should be identified [20,65].

Quality control emphasizes on testing to detect defects according to quality objectives and reporting to management to make the decision [6]. Thus, quality control is a process within quality assurance, whereas quality assurance goes beyond quality control with not only detecting but beside that, improve quality by avoiding or at least minimizing issues that lead to defects [6]. These general concepts about quality have led our thinking to introduce our framework for context quality assurance.

The quality management process should be not isolated from the context management process. Within

each CAS, context management framework is the part responsible for context management. The next subsection describes its function and structure.

6.2 Context Management Frameworks

The part within CASs, which is responsible of quality assurance and quality control, is the Context Management Framework (CMF). According to [63,66], CMF is responsible for main functions that affect context quality, i.e., collecting sensor data, aggregating that information to compose the context, and extracting high-level context information by performing reasoning operations. Authors in [63] named it as Context Monitoring and Management Frameworks (CMMF).

Authors in [66] introduced an approach for context management systems; this approach improves context information qualities and reduces overall performance cost for context-aware systems. It exploits context information and context metadata information for context management. Authors in [66] presented six elements for CMF: context aggregator service, context discovery service, context provider service, context observer service, context ontology reasoned service, and context query service.

CMF should implement a mechanism to verify and control quality of context information in order to improve decision-making support for context-aware system that belongs to [41]. Quality control is a part of quality aggregator element [11,63]. This component/element helps the management framework system selecting the high-quality context based on quality attributes and quality evaluation method applied. Authors in [63] named this element as QoC evaluator. QoC evaluator (controller) evaluates QoC parameters for a context to help CMF resolving conflict and redundancy problems. Earlier in this section, we described context conflict resolving as a vital quality control process in CASs.

Authors in [2] introduced a clear example for CMF. They introduced a layered conceptual framework. The first layer is called sensors layer; it consists of a collection of different sensors. The word "sensor" refers to every data source that may provide context information. According to that, sensors can be classified into three groups: (1) Physical sensors, which represent the most frequently, used type. (2) Virtual sensors: this type acquires context data from software applications or services. For example, it is possible to determine an employee's location by a virtual sensor, e.g., by browsing an electronic calendar, a travel-booking system, emails etc. (3) Logical sensors: these sensors use couple of information sources. They combine physical and virtual sensors with additional information from databases or various other sources in order to solve higher tasks.

For example, a logical sensor can be constructed to detect an employee's current position by analyzing logins at desktop PCs and a database mapping of devices to location information. The second layer is called raw data retrieval. It uses appropriate drivers for physical sensors and APIs for virtual and logical sensors. The query functionality is implemented in software components which make low-level details of hardware access transparent by providing more abstract methods such as getPosition(). By using interfaces for components, it is possible to replace an RFID system by a GPS system without any major modification in the current and upper layers. The third layer is pre-processing layer. The pre-processing layer is responsible for reasoning and interpreting contextual information. The sensors queries in the underlying layer most often return technical data that are not appropriate to use by application designers. This layer transforms the results of layer two to a higher abstraction level. In addition, in context-aware systems consist of different context data sources, the single context atoms are combined to high-level information in this layer.

This process is also called "aggregation" or "composition". Context conflicts that might occur when using different data sources has to be solved in this layer as well. Often this conflict is resolved by using additional data such as time stamps and resolution information. The fourth layer, Storage and Management, organizes gathered data and offers them via a public interface to the client. The Application layer is responsible for implementing the actual reaction of different events and context instances.

6.3 Resolving Context Conflicts in CASs

In CASs, a solution that provides a ubiquitous context assembles the context information from a group of related context services, which is called context fusion [20]. This fusion is also required because sensing technologies are not 100% reliable or deterministic [23]. Even for the same instance of context, multiple sensors are commonly used for dealing with context

ambiguity [18]. On the other hand, it is very often to find more than one user share the same ubiquitous context-aware application and the same resources, conflicts may occur during adaptation actions due to individuality.

An approach to deal with conflicts is to allow users to manually resolve the ambiguity in context. Rather than using an automated approach, this approach exploits a user's knowledge about the situation to help resolving and removing any ambiguity in the sensed or inferred context. A user may be provided by a list of the most likely interpretations of context, ranked by likelihood, and asked to select the "correct" interpretation [18]. However, this approach contrasts with the basic aspects of UC especially the invisibility, the natural interaction, and the autonomous systems. Context in nature has different levels, context conflict might occur through the different levels of context; while collecting sensor data from redundant context sources or while inferring the context situations in higher level or even when serving the different consumers of context. Literatures handled context conflicts for different context levels using different approaches. Authors in [67] distinguishes three categories of failures, they are: source failures, data failures, and context failures. With source failure, some sensors could be broken or lost connectivity, and no data is available. Data failures means that the reported value is completely out-of-range, presents abnormal variability, too little precision or is not updated frequently enough. The context failure means that the context modeling which is typically a statistical process, may introduce a particular hypothesis as the most probable one, but it does not match reality. The idea of quality policy is used for resolving conflicts. Authors in [68] introduced conflict resolving policies that are defined on the basis of the quality of context parameters. Remarkable solution introduced by authors in [69] defines three layers of conflicts to handle all types of conflicts and adopts three policies for conflict resolution in three different layers accordingly: source layer with freshness policy, processor layer (modelling and reasoning) with reliability policy, and consumer layer with priority policy.

The experiments proved that using a combination between freshness policy and reliability is better. Similarly, two types of conflicts are addressed by [70]: service resource conflict and user preference conflict. Service resource conflict can occur when selecting only some users among many users that want to be provided with the same service, this can happen due to limited service resource. User preference conflict occurs when providing only some users with personalized service, because the preference of users is not the same despite the same context of users is identified. These two problems are still not resolved perfectly, because it is difficult to provide personalized services under limited resources [5,27,70].

The next three subsections describe in details the effort that is spent by the research community to handle context conflicts based on the three layers proposed in [69]: source layer (sensed context), processor layer (derived context), and consumer layer.

6.4 Context Conflicts Resolving in Sensed Context Layer

When conflict occurs, a system has to choose one context among many conflicted contexts; selection of a context value should be done based on quality indicators or specific heuristics. Many QoC policy-based approaches were proposed in [1, 11, 40, 41, 47, 62, 63, and 71] for resolving sensed context conflicts.

Authors in [62] introduced an approach for resolving sensed context conflicts based on two views: quality policy and application requirements. Authors indicated that the solution should be applied by the context provider to protect the CAS from error propagation. As a part of a proposed middleware, authors in [47] proposed some quality attributes/indicators to cope with sensor limitations and discussed some alternatives to quantify them. These quality indicators are precision, freshness, spatial resolution, temporal resolution, and probability of correctness.

On the other hand, authors in [11] and [41] exploited the previous history of a context to forecast the correctness of the current one. The solution introduced by [11] and [41] resolves conflicts within the context fusion layer of a context management framework (CMF). This layer detects and resolves conflicts according to two quality attributes: probability_of_correctness and trustworthiness. Authors in [11] and [41] benefited the solution formalized for Byzantine Generals problem in [57]. Since, this problem is a trustworthiness problem. By applying this formalism to resolve conflicts problem, the number of context dimensions that affirm a context data under investigation should meet the proportion

of 3m + 1 to be considered reliable by the system, where m is the number of context dimensions that contradict it, i.e., at least two-third of context elements should affirm the investigated context element.

Authors in [11] classified conflicts into internal and external. They defined them as follows: internal conflict is "the context conflict/inconsistency that may occur by fusing two or more context elements that characterize the situation from different dimensions of a same observed entity in a given moment". Two context data (or more) are concerned in internal conflicts when CMF could not deduce which one is correct in fusion time. For example, let Tom is a user in a smart home system. The CMF indicates that Tom is in the bedroom based on WiFi-based location system. However, the light of bathroom is "on" and his agenda indicates that he has an appointment with his family doctor. In this case, there is an internal context conflict when fusing these three contexts to deduce current location of Tom.

Authors in [11] defined external conflicts as "the context conflict/inconsistency that may occur between two or more collected context data that describes the situation of an observed entity from the same point of view". For example, the indoor location of Tom determined by solutions such as WiFi, RFID, and Bluetooth technologies, i.e., each context data is gathered from a different context source to characterize the situation from the same context dimensions (location). In such case, which source should be selected to provide the correct data that composes current situation? This case is called external conflicts.

To resolve internal conflicts, the approach proposed in [11] is based on the idea that for a specific situation, the context element is not used alone; almost there is other context elements used along with it. There is a collection of context data that usually occurs together. Therefore, this approach utilizes the previous history of a context and dependencies between context elements to increase the probability that a certain context is correct.

The proposed technique uses Bayesian analysis to analyze the last occurrence of a piece of context and estimate probability of correctness of that piece of context. Context dependencies and relations affirm/contradict with probability rate within the range [0, 1] since in real situations, the degree of affirmation/contradiction is variable. To resolve the external conflicts, the proposed technique benefits the probability of correctness to compute trustworthiness using predetermined thresholds identified by the user.

Beside what mentioned in [11] and [41] about using forecasting for resolving context conflicts, many other techniques for forecasting can be also used.

6.5 Context Conflicts Resolving in Derived Context Layer

In this section, a state of the art on context conflicts resolution in higher level (context situation) will be articulated. Higher level context means there are conflicts with inferred/aggregated context.

Many literatures employed the idea of application policies to handle conflicts in higher level context [59, 60, 72-74]. Authors in [72], introduced an algorithm to resolve context conflicts for context situation in case that there are many policies for applying different context situations and these policies do not cover all context situations that could occur in real time. The proposed solution calculates the offset for each conflicted situation and all policies, and then chooses the nearest policy.

Authors in [73] proposed a new idea for using policies where they use different 8 ordered policies for resolving conflicts in the case that the current policy does not resolve the conflicts. Authors in [74] introduced a solution for detecting and resolving context conflicts for CASs authorization system using the idea of policies. Authors used context graph-static model for detecting the context conflicts.

In [59, 60], authors introduced a policy-based solution for resolving context conflicts with differentiation between two types of conflicts: static and dynamic conflicts. Static conflict can be resolved in compile time using one of predefined policies and dynamic conflicts is the conflicts which cannot be detected and resolved during the compile time. This situation could occur when the objectives of all active policies cannot be met. On the other hand, authors in [60] introduced a mechanism for assuring the consistency of the policies to avoid any conflict in the policies in case there is more than one policy applicable for one context situation.

New idea introduced in [17] by using fuzzy-logic based decision-making engine for high-level context analysis and conflict resolution.

A prevention approach introduced in [75] by avoiding context conflicts (detection before happening) and not only conflicts resolving. The

proposed solution is based on modelling context and modelling expected conflicts using semantic-rules and reasoning engine.

On the other hand, authors in [76] introduced a remarkable approach to deciding if context is predictable or not before using prediction for deriving context and consuming a lot of cost with unpredictable context. The proposed approach is based on the analysis of the time series representing the previous context information.

6.6 Context Conflicts Resolving in Consumer Layer

Despite that most context management frameworks/middleware manage situations where a single user exists at one time in a given context, it is very common and natural to assume that there are more than one user in real life environments such as home, office, etc. In these environments, users often compete against each other -either explicitly or implicitly- especially when they access the same resources. Many proposed solution used priority-based policy to handle context conflicts.

Authors in [28] proposed a solution that adopted the idea of assigning weight value for users' preferences to resolve context conflicts. In addition, authors introduced a mechanism for conflicts detection using semantics (ontology representation) without explicit descriptions of the conflicts between different applications. Authors in [19] utilized various factors such as priority, credits, age, and time when formalizing the policies.

On the other hand, authors in [5, 23, and 77] introduced an approach that makes trade-off between QoS/QoE and resource/service consumption; in other words, the adopted policy for resolving conflicts minimizes the cost. Quality of service means that users' satisfaction should be satisfied with the application's tasks.

Another approach introduced by authors in [61] for avoiding context conflicts between different applications by allowing them to define context situations that are considered to be conflicts. After detection, the system should try -at first- to resolve the conflict automatically. An initial approach used to resolve conflicts is simply done by banning the execution of the respective application.

A more complex strategy is the adaption which can be done by inducing a negotiation between conflicting applications. In [78], authors developed another prevention solution by introducing a solution that starts within a CAS design phase. For each provided activity, a software module should be designed. All modules developed for collective applications (for different users) are encapsulated into a single block called "conflict engine", which is performed during a single activity. The conflicts detection process performs a three dimensional analysis: involved users profiles, environment profile, and application tasks [78].

A preventive and proactive approach was introduced by authors in [26] using two mediated solutions. The first solution proposed a module that evaluates all users' feelings as a group and controls the mediation. The second solution used prediction to shorten the mediation time.

7. Discussion and analysis

As shown in this review paper, a lot of work has been done by the research community on QoC. We have remarked many points that reflect gap in the literature on QoC. These points can be summarized as follows:

Most work achieved in this area deals with context quality issues separately and not as integrated parts. There are many issues related to QoC that have been addressed in the literature such as context quality parameters [3,7,12,14,27,44,47,79,80], context quality measures [12,14,80], resolving context conflicts [11,40,63,66], context validity, and resolving context uncertainty [54-56,80]. However, realizing quality aspects needs to comprehensive solutions that can integrate all these issues together taking into account the different elements of QoC within CASs.

Many solutions, which are introduced in the literature, have been built depending on a simple view of the context, whereas the context has a complex taxonomy and different views. Thinking about QoC should start by thinking about the nature of the context itself. For example, context information combines different levels of context. In the low level, there are context elements or context facts, which are aggregated to compose a higher context information (abstracted context), and then abstracted contexts with each other compose the context situation in the top level. Despite this point of research is vital, very few studies have insufficiently addressed this issue [54-56].

To explain this point, let us introduce the following example: in flooding forecasting context-

aware system, there are wind speed, temperature, soil saturation, precipitation and rainfall duration which are considered context facts. Rainfall status is an abstracted context, which is concluded using rainfall duration and precipitation context facts. Rainfall status cooperates with the other abstracted contexts such as soil status to compose the final situation of the context, which is the "potential flooding". For this situation, quality of context in the lower level should affect quality of context in the upper level. To the best of our knowledge, this view has not been clearly addressed by the existing solutions of context quality.

On the other hand, a context fact could be sensed context such as temperature and user movement, or profiled context such as user calendar, whereas the context is derived in higher levels. In fact, quality aspects of sensed context should not be the same for profiled context.

Thus, these different views for context should be taken into account when thinking about any comprehensive solution to context quality control.

Most existing solutions of context quality parameters did not differentiate between the two basic types of parameters: basic quality parameters, which reflect context validity as a basic quality requirement (e.g. reliability, probability_of_correctness, freshness, and completeness), and the perfection parameters, which reflect other aspects of quality (e.g. privacy, precision, and representation_consistency). We believe that the basic quality parameters should take place, as it could be a basis for many context shortcomings such as context conflicts and context uncertainty.

Moreover, one shortcoming which can be addressed regarding QoC issue is that the previous approaches introduced for improving quality did not compare results against each other to evaluate their success; different researches used different criteria. For example, authors in [66] use the idea that the quality should reduce the computing cost; they used the cost as evaluation criteria. Number of computed triples against computing time is used to estimate quality improvement against cost. However, authors in [63] added number of context objects that have been deleted from the context store in a given period of time because they did not meet the quality criteria that have been set by user. To assess the success of quality solutions, many aspects should be tested. We are in need to determine the reflection of QoC on CASs and what are the criteria that should be used. Measures for quality control and end-user satisfaction of context-aware products need to be outlined [9].

Thus, we have to define and measure the overall performance using different quality parameters in terms of predefined criteria. In addition, we have to produce a wide scale model for QoC parameters for different applications using different parameters that are weighted based on application and context taxonomy. In the same point, the relations and interdependencies between these parameters should be addressed carefully to examine if we can exclude some of them to get more agile models of parameters.

Regarding to quality control processes, most work that has been achieved has focused on conflict resolving, maybe this is because other shortcomings such as missing values can be handled using general statistic and intelligent techniques. The proposed solutions for sensed context conflict resolving still have some shortcomings. Using QoC parameters only for conflict resolving is not enough. That is because of the shortage of sensors, which are the basic resources of context element values. Three basic problems can be addressed with sensors reliability:

(1) The default accuracy regarding to the sensor technology type. For example, the spatial position of an object could be captured using different technology (accuracy) such as GPS, infrared, and Bluetooth; each one of them has its default according to the technology.

(2) The distance between sensor and object. Each type of sensors has different spatial range for reliable sensing.

(3) Failures could happen with hardware due to different expected or unexpected reasons. These failures do not necessary make the sensor out of service; sensor can continue providing data but unfortunately with wrong values. This scenario could happen when a context-aware system is running thus; these systems could be critical and could lead to serious problems. We can select the best sensor according to first and second problems. This means that the higher value of reliability regarding the accuracy is defined by manufacture according to the distance between sensor and object. This could be great if the sensor works very well and does not suffer from any hardware failures. However, if the sensor has any undetected hardware problem, these two factors do not make a sense for the reliability.

Therefore, a solution that uses the previous context history to resolve conflicts could be a good

solution. It can exploit the previous history to calculate probability of correctness for each conflicted value. The method that can be used to calculate total probability of correctness is to use the average of conditional probability for context element under investigation given other context elements. Sometimes, the average leads to misleading results. In case of different affirmations of context elements have extremes, this will affect the average value. Thus, probability of correctness for some context elements could be better or worse while this is not correct. Summation of affirmation would be better, because we can know the maximum value of the affirmation if we know how to use context elements for prediction.

Beside quality parameters that indicate quality level of each piece of context, other general long-term quality indicators are essentially needed to serve different context stakeholders such as context provider, consumer, and CAS developers. To the best of our knowledge, most work that has been conducted by researchers has a lack regarding to this issue. For example, the quality level of a group of context facts in the lower level will introduce an important general indicator to know the quality of the sensor network that produced that context. These indicators could indicate the need to improve the whole network of sensors in a particular area. For example, a network of weather sensors belongs to a particular station that feeds a weather forecast context-aware system.

The next generation of context-aware ubiquitous systems is rapidly growing and in the near future it should be standardized for quality parameters and indicators, which requires a lot of integrated effort done by the research community.

8. Research trends in Context-Aware Systems

As a hot area, it is obviously to say that UC and CAS systems have many research trends in different aspects and several issues. From our point of view, these trends come mainly from the gap between the way the people are realizing, thinking and judging and the way that the computer does. Intensive and integrated research effort between the computing field and the medicine field should be planed and arranged carefully. The majority of research should be devoted for understanding the super abilities of human in thinking and realizing the context and the way they are judging and making decisions. Based on that, the abilities for the current CASs sensor-rich environment still so far and so silly. Existing researches recommended various research directions for context-aware systems as detailed below.

8.1 Producing High Quality Context

CASs are smart systems that make their decisions based on the context that they realize. It is "context-aware". According to author in [81], the phrase "context-aware" is misleading as this term indicates that CASs can sense and realize the real world around just as human do. These abilities are still very far from reality. The idea is not solely, in how the five human sense works together but also in how people thinking, interpreting, concluding and judging. Researches in machine learning algorithms and context fusion are still far away to simulate the human abilities [81]. Many different context integration, context reasoning and distribution techniques can be used to acquire and distribute context. Understanding context data and appropriately annotating it automatically forms a real problem [82]. Robust quality frameworks should be developed to accompany the trip of context from acquiring, fusion, integrating, deriving and distribution into making the different decisions based on this context. Context quality frameworks need to be defined and employed for different levels of context and different layers of CASs; developing an efficient uncertainty management models to capture the uncertainty aspect in CASs different operation is very important [82].

On the other hand, the nature of ubiquitous computing environment where CAS is laying adds other obstacles. Mobility, openness, interleaves systems are some examples for this environment. In addition, many sensors within CASs are displacement in motion-based activity recognition systems that actually affect the application services accuracy [83].

Many sub-trends can be derived from this direction such as sensors selection, context capturing, context fusion, context deriving, context quality, and context management frameworks. However, we prefer to use the "context quality" terminology because it can express and conclude the concept deeply. Quality of context should be concern at all levels of context production processes to create a true awareness for CASs. The benchmark of these processes that should be concerned is the human mental and emotional processes and the synergy between the two sides. The

quality of context, the quality of process and the quality of sensors/devices could be good triggers for research in this area.

8.2 Efficient Proactive Learning Algorithms for context processing

CASs is the basic technology of Ubiquitous Computing. According to authors in [84], the major aspect that can summarize CAS abilities is "Proactivity". To make CASs proactive, it should be smart, good learner, has a long memory and can learn from its experience. Creating like such environment is extremely important and not easy. To what extent the previous history can help in learning and predicting the current context [85]. Current context inference algorithms are still complex and heavy to deploy in UC on-line environment. Un-supervised algorithms could be good solution for producing lightweight context inference algorithms [83]. Actually, this research trend could be involved in the previous subsection as one of the main process for producing high quality context; however, we prefer to present it in a separated subsection due to its importance in UCs.

8.3 Security/Privacy/Trust

Within UC and every day computing environment, autonomous every day activities in our real life and the huge number of CASs around us will create a general feeling of users that they are being monitored all the time [83, 84]. Trust, security and privacy will be one important issue during designing the several layers CASs [83, 82].

8.4 Generic Context-Aware Frameworks

In ubiquitous computing environment, different CASs interact with each other, share the context and may share the sensors networks for different purposes. Sensor networks encompass different sensing technologies with different accuracy. Under what circumstances should CASs use one and not another. What are the minimal services that an environment must provide to make context awareness feasible? What are reasonable fall-back positions if an environment does not provide such services? [85]. Designing a generic CASs Frameworks and middleware will reduces the cost of handling contextual information selection based on its accuracy and the different purposes of CASs. Designing such CASs frameworks and standardization of CASs infrastructure has not clearly identified. The challenge is coming from the diversity of context-aware applications and their required types of contextual information, sensors, and inference algorithms types. One solution could be benefiting advantage of emergent cloud computing technologies. Cloud computing enables information sharing and other situational resources among mobile devices. Context information could be stored in a cloud and used by different CASs applications for different purposes.

Thereby, this information has does not need to be captured from different sensors networks that will be decrease the computation cost in general. The same sensor network could provide the same contextual information in different accuracy levels to be appropriate to the different needs of CASs applications. Different types of sensors and different types of inference algorithm could be used by trade-off between the context information accuracy and the time/power cost [85, 83, and 84].

8.5 Efficient Solutions

CAS usually works all the day thus, what is the overhead of considering context in a CAS? What techniques can we use to keep this overhead low? The energy/power consumption and computing cost should be one of the important research trends that need to be considered [85, 83]. CASs use a wide range of sensor networks and context inference algorithms with different level of efficiency.

Until now, using smart devices depends on a limited battery. The growth of energy is slow compared with the increasing need of energy consumption [83]. Efficient sensor management systems infusing low level sensory operations need to be addressed. Authors in [83] introduce some suggestions about that. The proposed solution is based on the idea of "dynamic sensor selection" where there are quantum and intervals for sensory sampling and selection. Thereby, sensors can be put in an order according to their power consumption levels and application relevance depending on an interested context [83]. On the other hand, establishing such models and software solutions for CASs power consumption in UC will be useful for dealing with this diverse environment where there are different sensors and devices with different energy consumption needs.

This could be achieved by finding the relationship between user activity and energy cost; special applications could be developed to gather data on user behavior by tracing usage pattern on device.

Current methods use external equipment to model energy estimation based on measurements of device operation in different operation modes. In contrast, recent studies prefer working without using extra equipment to track key operating system parameters and hardware components [85,83]. In addition, studying the relation between the usage patterns and battery depletion in non-linear mode will make CASs to be energy-aware and will lead to successful discovery of optimal energy reduction solutions that will eventually help maximize the wasted energy consumption to improve the QoS of CASs [83]. Besides battery issues, context inference algorithms are complex and heavy to process on-line as required in UC, there is a need for producing lightweight context inference algorithms [83].

8.6 CASs Services Using Mobile Cloud Computing

Mobile cloud computing is a growing research area that will improve the CASs performance. The idea is building a shared pool of context resources that are provisioned by cloud Internet services and mobile networks. This will be provided with minimal cost of power benefiting from the mobile cloud power and passing the limitation of traditional sensors and devices powered by batteries [83].

9. Conclusion

This paper articulates a survey of the state of the art on QoC in CASs. It presents an analytical review of QoC issues in CASs including QoC definitions, QoC parameters, quantification methods of QoC parameters, and resolving context conflicts in CASs. The paper then discusses the current and future research directions on QoC in CASs. It is argued that a comprehensive framework can be developed to overcome the limitations of current QoC models in CASs; this will be addressed in future work.

References

[1] P. Gabriel, M. Bovenschulte, E. Hartmann, W. GroB, H. Strese, K. M. Bayarou, M. Haisch, M. MattheB , C. Brune, H. Strauss, H. Kelter and R. Oberweis, “Pervasive Computing: Trends and Impacts, Editorial: Federal Office for Information Security”, Publisher: SecuMedia ISBN 3-922746-76-4, 2006.

[2] M. Satyanarayanan, “Pervasive Computing: Vision and Challenges”, Personal Communications, IEEE, Volume 8, Issue 4, Page(s):10- 17, 2001.

[3] J.Zhao Sun, “Mobile Ad Hoc Networking: An Essential Technology for Pervasive Computing”,International Conferences on Info-tech & Info-net, Beijing, China, 2001.

[4] M. Lethrech, I. Elmagrouni, M. Nassar and A. Kriouile, “Domain Specific Modeling Approach for Context-aware Service Oriented Systems”, 2014 International Conference on Multimedia Computing and Systems (ICMCS), IEEE Proceedings, Marrakech, 14-16 April 2014.

[5] K. Mitra, A. Zaslavsky and C. Ahlund, “Context-Aware QoE Modelling, Measurement, and Prediction in Mobile Computing Systems”, IEEE Transactions on Mobile Computing, Vol. 14, Issue 5, pages 920-936, May 2015.

[6] J. Berri, “Context reasoning for mobile systems”, 2014 World Congress on Computer Applications and Information Systems (WCCAIS), IEEE Proceedings, 2014.

[7] A. Manzoor, H. Truong and S. Dustdar, "QoC: Models and Applications for CAS in Pervasive Environments", *Special issue on Web and Mobile Information Services*, **2010**.

[8] A. Manzoor, H. Truong and S. Dustdar, "On the Evaluation of QoC", Proceedings of the 3rd European Conference on Smart Sensing and Context, Springer-Verlag Berlin, Heidelberg, **2008**.

[9] J. B. Filho and N. Agoulmine, "A quality-aware approach for selecting context information from redundant context sources", 7th Network Operations and management Sumposium, IEEE Proceedings, 2011.

[10] K. Henricksen and J. Indulska, "Modeling and using imperfect context information", Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, IEEE, 2004.

[11] H. Hegering, "Management Challenges of Context-Aware Services in Ubiquitous Environments", DSOM 2003, LNCS 2867, IFIP International Federation of Information Processing, 2003.

[12] M. Krause and I. Hochstatter, "Challenges in Modelling and Using Quality of Context (QoC)", T. Magedanz et al.(Eds.): MATA 2005, LNCS 3744, pp. 324–333, Springer-Verlag Berlin Heidelberg, 2005.

[13] Y. Bu, T. Gu, X. Tao, J. Li, S. Chen and J. Lu, "Managing QoC in Pervasive Computing", Sixth International Conference on Quality Software (QSIC'06), 2006.

[14] B. Thomas and M. Schiffers, "QoC: What It Is And Why We Need It", In Proceedings of the 10th Workshop of the Open View University Association: OVUA’03 , 2003.

[15] S. Singh, S. Puradkar and Y. Lee, “Ubiquitous Computing: Connecting Pervasive Computing through Semantic Web”, Information Systems and e-Business Management Journal, Volume 4, Issue 4, pp 421-439, Springer, 2006.

[16] M. Friedewald and O. Raabe, “Ubiquitous computing: An overview of technology impacts”, Journal of Telematics and Informatics, Volume 28, Issue 2, Pages 55–65, Elsevier, 2011.

[17] C. Silva and M. A. R. Dantas, “Quality-Aware Context Provider: A filtering approach to context-aware systems on ubiquitous environment”, IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2013.

[18] B. KANG, “Ubiquitous Computing Environment Threats and Defensive Measures”, International Journal of Multimedia and Ubiquitous Engineering, Vol. 2, No. 1, January, 2007.

[19] V. Tuttlies, G. Schiele and C. Becker, “COMITY - Conflict Avoidance in Pervasive Computing Environments”, In Proceedings of the Second International Workshop on Pervasive Systems (PerSys '07), Vilamoura, Algarve, Portugal, November 25 - 30, 2007.
[20] Y. Lu, M. Motani and W. C. Wong, “A QoE-Aware Resource Distribution Framework Incentivizing Context Sharing and Moderate Competition”, Transactions on Networking, IEEE/ACM, Volume: PP, Issue: 99, pages 1-14, March 2015.
[21] http://www.ubiq.com/hypertext/weiser/UbiHome.html.
[22] A. Galloway, “Intimations of everyday life”, Ubiquitous Computing and the city Cultural Studies, Vol 18, Issue No. 2-3, pp. 384–408, 2004.
[23] G. S. Thyagaraju, S. M. Joshi, U. P. Kulkarni, S. K. M. Narasimha and A. R. Yardi, “Conflict Resolution in Multiuser Context-Aware Environments”, IEEE CIMCA 2008.
[24] Position Classification Standard for Quality Assurance Series, GS-1910. US Office of Personnel Management. March 1983. Retrieved 21 December 2012.
[25] R. Jason Weiss and J. P. Craiger, “Ubiquitous Computing”, Leading edge, Volume 39, Number 4, April 2002.
[26] M. ElGammal and M. Eltoweissy, “Towards Aware, Adaptive and Autonomic Sensor-Actuator Networks”, Fifth IEEE International Conference on Self-Adaptive and Self-Organizing Systems, 2011.
[27] H. A. Edelstein, “Introduction to Data Mining and Knowledge Discovery”, Third Edition, Two Crows Corporation, ISBN: 1-892095-02-5, 1999.
[28] R. Rao, G. Ye and J. You, “HCAM:A Context-aware Middleware to Support Logic-based Context Conflict Detection”, Research Article in 2nd International ICST Conference on Communications and Networking in China, 2007.
[29] Prakriti Trivedi, Kamal Kishore Sagar, Vernon, “Emerging Trends of Ubiquitous Computing”, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 1, No.3, September, 2010.
[30] Lathies Bhasker, “Pervasive Computing Issues, Challenges and Applications”, International Journal Of Engineering And Computer Science, Volume 2, Issue 12, Pages 3337-3339, December 2013.
[31] Gregory D. Abowd and Elizabeth D. Mynatt, “Charting Past, Present, and Future Research in Ubiquitous Computing”, Journal ACM Transactions on Computer-Human Interaction (TOCHI), Volume 7 Issue 1, Pages 29-58, March 2000.
[32] Roy Want, Trevor Pering, Gaetano Borriello, Keith I. Farkas, “Disappearing Hardware”, Journal IEEE Pervasive Computing, Vol. 1, Issue 1, Pages 36-47, January 2002.
[33] Mehdi Mekni, Andr´e Lemieux, “Augmented Reality: Applications, Challenges and Future Trends”, International Journal of Advanced Research in Computer Science, Volume 5, No. 7, October 2014.
[34] Nigel Davies, Hans-Werner Gellersen, “Beyond Prototypes: Challenges in Deploying Ubiquitous Systems”, Journal IEEE Pervasive Computing, Volume 1 Issue 1, Pages 26-35, January 2002.
[35] Anuj Aggarwal, Sunil Kr Singh, Kavneet Kaur, “Emerging Trends and Limitations in Technology and System of Ubiquitous Computing”, International Journal of Advanced Research in Computer Science 5 (7), 174-178, 2014.
[36] M. Satyanarayanan, “Privacy: The Achilles Heel of Pervasive Computing?”, Journal IEEE Pervasive Computing, Volume 2 Issue 1, Pages 2-3, January 2003.
[37] ISO 9000:2005, Clause 3.2.10
[38] D. Adsit. "What the Call Center Industry Can Learn from Manufacturing: Part I". National Association of Call Centers, 2007, Retrieved 21 December 2012.
[39] D. Adsit. "What the Call Center Industry Can Learn from Manufacturing: Part II". National Association of Call Centers, 2007, Retrieved 21 December 2012.
[40] J. B. Filho and N. Agoulmine, "A Quality-Aware Approach for Resolving Context Conflicts in Context-Aware Systems", In proceeding of IEEE/IFIP 9th International Conference on Embedded and Ubiquitous Computing, EUC 2011, Melbourne, Australia, 2011.
[41] F. Paganelli, G. Bianchi and D. Giuli, "A Context Model for Context-aware System Design towards the Ambient Intelligence Vision: Experiences in the eTourism Domain", Universal Access in Ambient Intelligence Environments 2006.
[42] A. A. Al-Shargabi and F. Siewe, "Resolving Context Conflicts Using Association Rules (RCCAR) to Improve Quality of Context-Aware Systems", ICCSE 2013, IEEE conference, 2013.
[43] J. B. Filho, A. D. Miron, and I. Satoh, "Modeling and Measuring Quality of Context Information in Pervasive Environments", 2010 24th IEEE International Conference on Advanced Information Networking and Applications, 2010.
[44] A. Toninelli, A. Corradi and R. Montanari, "A Quality of Context-Aware Approach to Access Control in Pervasive Environments", The Second International ICST Conference on Mobile Wireless Middleware, Operating Systems, and Applications, Berlin, 28 April 2009.
[45] N. Brgulja, R. Kusber, K. David, and M. Baumgarten, "Measuring the Probability-of-Correctness of Contextual Information in Context Aware Systems", DASC 2009, IEEE Conference, 2009.
[46] R. Neisse, M. Wegdam and M. Van Sinderen , "Trustworthiness and Quality of Context Information" ICYCS 2008, IEEE Conference, 2008.
[47] H. Kerzner, “Project Management: A Systems Approach to Planning, Scheduling, and Controlling”, John Wiley, ISBN 978-0-470-50383-6, 2009.
[48] Z. Abid and S. Chabridon, "A fine-grain Approach for Evaluating the Quality of Context", Pervasive Computing and Communication Workshop PERCOM 2011, IEEE, 2011.
[49] K. Sheikh, M. Wegdam and M. V. Sinderen, "Middleware Support for Quality of Context in Pervasive Context-Aware Systems" ,Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops(PerComW'07), 2007.
[50] S. McKeever, J. Ye, L. Coyle and S. Dobson, "A multilayered uncertainty model for context aware systems", The international conference on Pervasive Computing, 2008.
[51] M. Mühlhäuser and I. Gurevych, "Introduction to Ubiquitous Computing", Handbook of Research: Ubiquitous Computing Technology for Real Time Enterprises, IGI Global, 2010.
[52] S. Poslad, "Ubiquitous Computing: Smart Devices, Environments and Interactions", John Wiley & Sons Ltd, 2009.
[53] K. Henricksen, J. Indulska, and A. Rakotonirainy, "Modelling Context Information in Pervasive Computing Systems", Springer-Verlag Berlin Heidelberg 2002, Pervasive 2002, LNCS 2414, pp. 167–180, 2002.
[54] J. Pike and R. Barnes, “TQM in action: a practical approach to continuous performance improvement”, ISBN 0-412-71530-9, 1998.
[55] K. R. Grimes, “ISO 9001:2000: a practical quality manual”, ISBN 0-87389-555-X, 2002.

[56] N. J. Jeon, C. S. Leem, M. H. Kim and H. G. Shin, “A taxonomy of ubiquitous computing applications”, Wireless Personal Communications, Volume 43, Issue 4, pp 1229–1239 , Springer, 2007.
[57] L. Lamport, R. Shostak and M. Pease, "The Byzantine Generals Problem", SRI International ACM Transactions on Programming Languages and Systems, Vol. 4, No. 3, 1982.
[58] E. C. Lupu and M. S. Sloman, “Conflicts in Policy-Based Distributed Systems Management”, IEEE Transactions on Software Engineering - Special Issue on Inconsistency Management, 1999.
[59] A. Masoumzadeh, M. Amini, and R. Jalili, “Conflict Detection and Resolution in Context-Aware Authorization”, 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07), IEEE, 2007.
[60] N. Dunlop, J. Indulska and K. Raymond, “Methods for conflict resolution in policy-based management systems”, 7th IEEE International Enterprise Distributed Object Computing, Conference, pages 98–109, Brisbane, Australia, IEEE Computer Society, 2003.
[61] L. Kagal, T. Finin and A. Joshi, “A Policy Language for a Pervasive Computing Environment”, IEEE 4th International Workshop in Policies for Distributed systems and networks, Italy, June, 2003.
[62] T. R. M. B. Silva, “On the Resolution of Conflicts for Collective Pervasive Context-Aware Applications”, IEEE, 2010.
[63] A. Manzoor, "Quality Aware Context Information Aggregation System for Pervasive Environments", WAINA 2009, IEEE Conference, 2009.
[64] A. K. Dey and G. Abowd, "Towards a Better Understanding of Context and Context-Awareness", Conference on Human Factors in Computing Systems, 2000.
[65] K. Nikita, “Context-Aware Sensing and Multisensor Fusion”, Wiley-IEEE Press, 2014.
[66] M. A. Razzaque, Simon Dobson and Paddy Nixon, "Categorization and Modeling of Quality in Context Information", CiteseerX, 2006.
[67] P. Fleury, J. Kleindienst and R. Kessl, "On Handling Conflicting Input in Context-Aware Applications", IBM Voice Technologies, Prague, Czech Republic, MLMM, CiteSeerX, 2005.
[68] A. Manzoor, H. Truong and S. Dustdar, "Using Quality of Context to Resolve Conflicts in Context-Aware Systems", QuaCon 2009, Springer, 2009.
[69] I. Park, D. Lee and S. J. Hyun, “A Dynamic Context-Conflict Management Scheme for Group-aware Ubiquitous Computing Environments”, Proceedings of the IEEE 29th Annual International Computer Software and Applications Conference (COMPSAC’05), 2005.
[70] J. E. Mcgonigal, “This might be a game: ubiquitous play and performance at the turn of the twenty-first century”, Doctoral Dissertation, University of California, Berkeley, CA, USA 2006.
[71] J. Ye, S. McKeever, L. Coyle, S. Neely and S. Dobson, "Resolving Uncertainty in Context integration and Abstraction", ICPS’08, ACM, 2008.
[72] John Krumm, “Ubiquitous Computing Fundamentals”, Taylor and Francis Group, LLC, ISBN 978-1-4200-9360-5, 2010.
[73] X. Yang, “An Adaptive Mechanism for Inconsistent Context Resolution in Ubiquitous Computing”, International Conference on Control Engineering and communication Technology, 2012.
[74] A. A. Alzahrani, S. W. Loke1 and H. Lu, “Conflict and Interference Resolution for Physical Annotation Systems”, 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 2013.
[75] T. R. M. B. Silva, L. B. Ruiz and A. A. F. Loureiro, “Solving Collective Conflicts in a Disaster Emergency Attendance Application”, IEEE Symposium on Computers and Communications, 2009.
[76] M. Musolesi and C. Mascolo, "Evaluating Context Information Predictability for Autonomic Communication", Proceeding WOWMOM '06 Proceedings of the 2006 International Symposium on World of Wireless, Mobile and Multimedia Networks, IEEE , 2006.
[77] Y. Qi, M. Xi, S. Qi and J. Zhao, “A Conflict Resolution Method in Context-Aware Computing”, 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007), 2007.
[78] T. R. M. B. Silva, A. A. F. Loureiro and L. B. Ruiz , “A Conflict Resolution Methodology for Collective Ubiquitous Context-Aware Applications”, Proceedings of the 13th International Conference on Computer Supported Cooperative Work in Design, 2009.
[79] Y. Kim and K. Lee, "A quality measurement method of context information in ubiquitous environments", In Hybrid Information Technology, ICHIT’06, Vol. 2, International, 2006.
[80] D. Hoyle, “ISO 9000 quality systems handbook”, ISBN 0-7506-6785-0, 2005.
[81] Sagar Sukode, Shilpa Gite, Himanshu Agrawal, "CONTEXT AWARE FRAMEWORK IN IOT: A SURVEY", International Journal of Advanced Trends in Computer Science and Engineering, Volume 4, No.1, January – February 2015.
[82] Thomas Erickson, "Some Problems with the Notion of Context-Aware Computing", COMMUNICATIONS OF THE ACM February 2002/Vol. 45, No. 2, 2002.
[83] Ozgur Yurur , Chi Harold Liu, Zhengguo Sheng ,Victor C. M. Leung, Wilfrido Moreno, and Kin K. Leung, "Context-Awareness for Mobile Sensing: A Survey and Future Directions". IEEE Communications Surveys & Tutorials, 2013.
[84] Akihirio Fujii, "Trends and Issues in Research on Context Awareness Technologies for a Ubiquitous Network Society", Information and Communications Research Unit, QUARTERLY REVIEW No.26 /January 2008. SCIENCE & TECHNOLOGY TRENDS, Issues for context awareness technology, 2008.
[85] M. Satyanarayanan, "Challenges in Implementing a Context-Aware System", Pervasive Computing, IEEE, 2002.
[86] H. Lee, J. S. Choi and R. Elmasri, "A Classification and Modeling of the Quality of Contextual Information in Smart Spaces", PERCOM 2009, IEEE International Conference, 2009.