ICIT 2015 The 7th International Conference on Information Technology
doi:10.15849/icit.2015.0111 © ICIT 2015 (http://icit.zuj.edu.jo/ICIT15)

The Dualism of Context in Ubiquitous Computing

Dennis Lupiana
Faculty of Computing, Information System and Mathematics
Institute of Finance Management
Dar Es Salaam, Tanzania

Fredrick Mtenzi
School of Computing
Dublin Institute of Technology
Dublin, Ireland

Brendan O'Shea
School of Computing
Dublin Institute of Technology
Dublin, Ireland

***Abstract*— Context-aware systems are fundamental for making the use of computing devices intuitive. These systems respond to their environments to facilitate seamless interactions between the users and their computing devices, and to make these devices less intrusive. Although it is more than a decade since context-aware systems were introduced, context is still not well understood within a context-awareness research community. Although there are numerous definitions of context, these definitions refer to context as an input or as a derivable. The majority of researchers believe any input that makes a context-aware system to accomplish its task is a context. In the contrast, there is handful of researchers who believe context is derived from more than one inputs. This paper aims to provide a clear meaning of context and consequently to resolve the differences between researchers regarding context. In particular, this paper answers the most fundamental, but yet the most avoided, question; w*hat is context*?**

Keywords— context, situation, context-awreness system; ubiquitous computing; context-aware architecture

I. INTRODUCTION

The Ubiquitous Computing (UbiComp) paradigm has inspired the invention of numerous computing devices. Although these devices offer the users many convenient ways of accomplishing their everyday tasks, it remains a challenge for the users to use them effectively. This is more challenging as these devices are mobile. The users' working environments become open and hence less predictable. The users and their devices enter and leave different working environments where different settings and computing needs may be required. This makes interacting with devices difficult and more time consuming.

In response to these challenges, a Context-Awareness research strand emerged. The main focus of this strand is to investigate different principles, methodologies and techniques required to develop software systems that can *adapt* to their dynamic environments and the users' computing needs [1]. These systems are called *context-aware systems*. Initial context-aware systems used location or identity information to automatically provide users' computing needs. To date there are many context-aware systems, each exploiting different aspects of the real world.

Central to a context-aware system is *context*. Despite of its importance, context is still not well understood within the context-awareness research community. As a result, context means differently to different researchers. Although there are numerous definitions of context, there are two notable interpretations of context; *context as an input* and *context as a derivable*. The majority of researchers believe any input that makes a context-aware system to accomplish its task is context. In the contrast, there is handful of researchers who believe context is derived from more than one inputs.

This paper aims to provide a clear meaning of context and consequently to resolve the differences between researchers regarding context. In particular, this paper answers the most fundamental, but yet the most avoided, question; *what is context*? This is not the first time this question is raised. Dey and Abowd [2] and Zimmermann and his colleagues [3] have raised and attempted to address this question. The majority of researchers avoid defining context and instead adopts the

doi:10.15849/icit.2015.0111 © ICIT 2015 (http://icit.zuj.edu.jo/ICIT15)

existing definitions of context. As contended by [4], defining context is difficult and hence researchers prefer to adopt the existing definitions. Others argue that the definition of context is not important but how context is used is. Since context is central to context-aware systems, in this paper we argue that a clear understanding of context is important.

The rest of this paper is organised as follows. The background of context-awareness computing is provided in section II where the reason for the different interpretations of context is outlined. The explanation of the two notable interpretations of context is provided in section III and IV. The discussion of these interpretations is provided in section V. Section VI discusses the implications and consequences of these two interpretations. Section VII provides a conclusion of this paper and the future work.

II. Background on Context-awareness

The pioneers of Context-Awareness computing [1] define context-aware system as a computing system that examines and reacts to individual's context. To examine is to scrutinise or analyse while to react is simply to respond to something. Hence, according to Schilit and his colleagues, context-aware systems are computing systems that analyse someone's context before responding to it. This implies that context is dynamic and thus context-aware systems should be adaptive to these dynamics. As Schilit and his colleagues assert, the constantly changing execution environment is a significant aspect of context-awareness. This leaves us with many questions but the most important one is; *what is context*?

Unfortunately, Schilit and his colleagues provide us with no definition of context. Instead, they assert that where you are (location), who are you with (other people) and what resources are nearby (accessible devices) are important aspects of context. As Schilit and his colleagues argue, the little information covering someone's proximate environment is the most important in context-awareness. Clearly, these aspects are "ingredients" and context is an end product. Hence, context-aware systems should use these "ingredients" as inputs to determine and subsequently to respond to someone's context. This implies that context-aware systems should be responsive to context and not to individual inputs.

In the contrary, initial context-aware systems [5-9] are described to be responsive to implicit inputs such as identity oand location. Weiser [10], for instance, describes a system that opens a door to the right badge wearer. In these systems, responses are predetermined and hence inputs are used as *cues*. Although Schilit and his colleagues [11] later call for a broader view of context, the description of the initial context-aware systems had already caused a considerable divide among researchers. While the majority of researchers believe that an input to a context-aware system is a context, few argue that context is derived from more than one input.

III. Context As An Input

Dey and Abowd [2] define a context-aware system as a computing system capable of providing information and/or services relevant to the user's task. This definition raises a question as to how would a context-aware system know what task a user is involved in? According to [2], context-aware systems do not know users' tasks but are programmed to provide relevant information and/or services in a task. To automate the latter process, these systems are developed to respond to cues. In a context-aware tour guide system, for example, a tourist is provided with relevant information about a site when approaching the site. In this example, the location of the site is a cue.

Dey and Abowd [2] refer to cues as context and define it as *any* information that can be used to characterise a situation of an entity, where an entity can be a user or any other object. This definition implies that context can be one or a set of inputs. In the tour guide example, for instance, location is a context because it is used to characterise the situation of the user. This is a fairly reasonable definition of context since it is open-ended. It provides flexibility to system developers to enumerate contexts as required by their systems. According to their example, a situation is a task that a user needs to accomplish. Hence, any information that is necessary for the system to accomplish this task is context. Although this definition is broad, as argued by [3], it addresses the criticism that it is impossible to enumerate exhaustive list of context.

Although it is more than a decade since this definition was proposed, it is still used in the recent work. Soylu and his colleagues [12], Liu [13] and van de Westelaken and his colleagues [14], for instance, used this definition. As stated by [4], defining context is difficult and hence the majority of researchers prefer to adopt existing definitions. Although few researchers have attempted to define context, in many cases these definitions are variations of the definition provided by [2]. Chen and Kotz [15], for instance, define context as a set of environmental states and settings. Similarly, Chen [16] defines context by replacing 'any information' from the definition provided by [2] with a list of attributes and entities within a physical environment. Zimmermann and his colleagues [3] attempt to narrow the definition provided by [2].

In this interpretation, context is regarded as an input to a context-aware system. In many cases, context is referred to an attribute of an entity, which is essential for a context-aware system to accomplish a specific task. Consequently, as shown in figure 1, context-aware system is regarded as a system that monitors (P_1) its environment and responds (P_2) to inputs from its sensors.

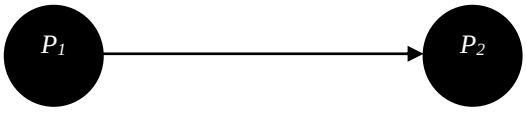

Fig. 1. Key processes of context-aware systems in this category

doi:10.15849/icit.2015.0111 © ICIT 2015 (http://icit.zuj.edu.jo/ICIT15)

Two concepts from the description of context-aware systems from [1] and [2], however, are not blending well into this interpretation of context. First is the analogy of human perception and information within someone's proximate environment. Human being uses information around them to understand their environments and hence to respond appropriately. Context-aware systems in this category, however, respond to individual pieces of information. Hence it is unclear on how context-aware systems use information from their surroundings to respond appropriately. Second is the idea of dynamism of context and how it is used in context-aware systems of this category.

IV. CONTEXT AS A DERIVABLE

"*In this model computation does not occur at a single location and in a single context, as in desktop computing, but rather spans a multitude of situations and locations* ..." [1].

It is evident from the excerpt that context is dynamic and it occurs in a location. This excerpt implies that location is not a context but it is part of context. As noted by [1], the dynamic environment of devices is the driving factor for designing context-aware systems.

Respond and *adapt* are two different terms and hence they should be carefully used, especially in context-awareness. In English, to respond means to act on return while to adapt means to modify or adjust to new conditions. Thus, in a responsive system the relation between inputs and outcomes is binary; the outcomes depend on whether the inputs exist or not. To be adaptive, however, a system should have a certain degree of correctness. This implies that an adaptive system should be able to examine or analyse its inputs. As noted by [2], "adapting to context" means a context-aware system can modify its behaviours accordingly.

Chen and Kotz [15] and Kaenampornpan [17] argue that a context-aware system should combine various inputs. Kofod-Petersen [18] also argues that if a context-aware system is unable to *reason* about various inputs, it cannot be adaptive to its context. Likewise, [12] argue that a context-aware system should exhibit intelligence. This implies that context is dynamic and hence context-aware systems should be able to contemplate different aspects of their environments before responding. This view of context-aware systems correlates with the definition of context-aware systems provided by [1]. These systems do not respond to individual inputs but after analysing these inputs. Hence context in these systems is derived from more than one input. Consequently, as shown in figure 2, a context-aware system is regarded as a system that monitors (P_1) its environment, analyses (P_2) its inputs and responds (P_3) to ongoing context. Hence a change of an input implies a change to an ongoing context. Thus, the dynamism of context is subject to different aspects of an environment.

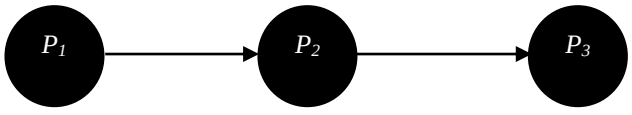

Fig. 2. Key processes of context-aware systems in the 'context as derivable' category.

This also explains why human perception and information within someone's proximate environment is relevant in context-awareness. Human perception can be summarized as a three-phase process, involving sensing of surroundings, interpreting of stimulus from sensory organs, and inferring what is going on. To know what is going on, human beings use their past experience about a phenomenon. In context-awareness, a similar argument has been raised. Bolchini and his colleagues [19], for instance, argue that context-awareness involves applying past experience to the available facts within the environment to understand what is happening. Similarly, [17] argues that a context-aware system should possess prior knowledge about situations. Therefore, like human beings, context-aware systems should also apply previous knowledge about contexts within their environments.

V. DISCUSSION

As explained in section III and IV, the term context has two notable interpretations in the context-awareness research community. On one camp context is referred to as any input to a context-aware system while on the other camp context is referred to as a product of a context-aware system after combining more than one input. One camp argues that any information is a context as long as it affects how a context-aware system operates. Hence, depending on a context-aware system, context can be one or more pieces of information. In contrast, the other camp argues that context is derived after analysing more than one input. In this camp context is a collective term used to describe circumstances of a user in the real world environment.

Let us assume that we have two context-aware systems; one that automatically opens a door to a right badge wearer and the other that remotely switches ON a user's computer when a user enters a room. Both of these systems depend on one input, which is the identity of the users' badges, but react differently. The user's goal in the first system is to open a door while the user's goal of the other system is to switch ON her/his computer when enters a room. Hence each of these systems responds to a user's goal. ID of the badge is used by these systems as a *cue*; for automating the process of opening the door or switching ON a computer. Thus, a context-aware system is developed to respond to user's goals and not to the inputs of the system.

Since context-aware systems respond to users' goals, these goals can be referred to as context. This type of context, however, is static as it is predefined by developers of context-aware systems. Consequently, as [2] found a decade ago, the majority of the existing context-aware systems are *responsive* rather than *adaptive* to their environments. Recently, we also

doi:10.15849/icit.2015.0111 © ICIT 2015 (http://icit.zuj.edu.jo/ICIT15)

arrived to similar findings [23]. The difference between these is that currently context-aware systems respond to more than one input. Hence, if a context-aware system is developed to automatically open a door to a right badge wearer, this system will open the door even when the badge wearer is not intending to enter the room. Hence, like the initial context-aware systems, these systems use inputs as cues.

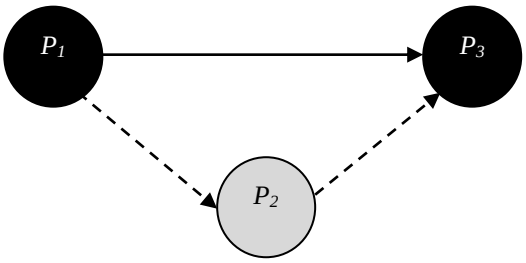

Fig. 3. Key processes of context-aware systems.

From the definition by [1], context-aware system should be capable of *monitoring (P_1), analysing (P_2)* and *responding (P_3)* to context as shown in figure 3. As asserted by [1], the constantly changing execution environment is a significant aspect of context-awareness. Hence all these processes play important roles but P_2 is the most important one as it determines context of a user. Thus, if at time T_t a system responded to one context then through P_2 the system would be able to respond to a different context at time T_{t+n}. Pieces of information gathered by the system, through P_1, is used as an input to analyse the individual's context. The majority of the existing context-aware systems, however, are developed with predefined context and subsequently without P_2. As a result, these systems cannot adapt to changes that occurs within their environments.

VI. IMPLICATIONS AND CONSEQUENCES

A. Inputs to Context-Aware Systems are Context Parameters

It is evident from the discussion that inputs to context-aware systems are parameters to these systems. Hence we argue that any information required by a context-aware system is a context parameter. We define context parameter as a piece of meaningful information that has an impact on a context-aware system. This information may be interpreted from data captured by a sensor or acquired directly from other sources such as a network or application software. The name of the owner of a device interpreted from the device's ID captured by a sensor, for instance, is a context parameter.

A widely used synonym of a context parameter is contextual information. This term, however, is interchangeably used with singular and plural meaning. Gu and colleagues [24] and Chen [16], for instance, refer to identity, location or time as contextual information while [25] and [26] refer to a set of context parameters as contextual information. Hence, to avoid this confusion, we prefer to use the term context parameter.

B. Context is What a System is Programmed to Do

It is evident from the discussion that context-aware systems respond by accomplishing whatever are programmed to do. Inputs are used by context-aware systems as cues to automate whatever task a context-aware system is programmed to do. To avoid contradictions with context, as it is widely used, [15] refer to this kind of context as a *high level* context. Gellersen and colleagues [20] refer to this kind of context as a *situational context*. Barkhuus [21] refers to this kind of context as a human context. Similarly, [22] refers to this kind of context as *activity* or *situation*. Likewise, we [23] refer to this kind of context as *situation*.

This paper adopts the definition of a situation from [18] who defines a situation as a social setting, such as a meeting, where the users involved want to achieve various goals. This definition of situation differs from that of [2], [26] and [27] as is not confined to a particular task. This definition emphasises meaningful interactions between relevant entities required to sufficiently describe the real world environment that is of interest to the users and their devices. A situation provides a detailed picture of the real world environment whereas a context parameter provides an aspect of the real world environment.

C. Architectural Support are Key to Context-Awareness

As [2] and [23] found, currently the majority of the existing context-aware systems are responsive rather than adaptive to their changing environments. These systems respond appropriately only to contexts that are programmed with. In most cases, any change within a physical environment does not affect how these systems should respond. For instance, a context-aware system that automatically displays presentation slide will continue to display a slide with sensitive information even if an unauthorized person enters a room. Hence, should developers change how context-aware systems are currently implemented to accommodate the adaptive nature of context-aware systems?

For more than two decades context-aware systems have been implemented for a specific purpose. As the pioneers of context-aware computing have outlined, these systems provide and/or gather some sort of information or services or trigger some actions on their host devices. Therefore, changing how these systems should be implemented runs counter to the very fundamental principle of their existence. Developers can attempt to implement general purpose context-aware systems but they should be aware that there is no a killer-application.

The best that researchers can do is to develop knowledge-driven context-aware architectures. Instead of context-aware architectures to be *passive*, as the majority, these architectures should be *active*. In addition to acquiring and translating data from sensors and other sources, these architectures should be able to use numerous context parameters to determine ongoing

doi:10.15849/icit.2015.0111 © ICIT 2015 (http://icit.zuj.edu.jo/ICIT15)

context. Hence, instead of context-aware architectures to be sharing context parameters with context-aware systems, these architectures should be sharing knowledge of what is happening. This will significantly increase application of context-awareness because even the most miniature and resource-limited devices would be made aware of their ongoing contexts.

To walk the walk, we have developed one of such architectures in the School of Computing at the Dublin Institute of Technology. In addition to monitoring and interpreting data gathered by sensors, our architecture called Knowledge-driven Distributed Architecture (KoDA) reasons about available information to recognise ongoing context. With KoDA, the implementation of context-aware systems is significantly simplified as they can be implemented without inference rules. This frees developers from the hurdles of learning knowledge representation languages required to represent inference rules. It also removes redundancy of inference rules as in KoDA these rules are centralised.

VII. CONCLUSION

In this paper we have review relevant literature in context-awareness computing with the aim of providing a clear meaning of context and consequently to resolve the differences between researchers. In particular, this paper answers the most fundamental, but yet the most avoided, question; *What is context*?. We categorised the existing definitions of context into the two notable interpretations of context; context as an input and context as a derivable. The 'context as an input' interpretation of context qualifies individual pieces of information that have effect on how a context-aware system operates to context. In contrast, the 'context as a derivable' interpretation of context refers to context as to what that is derived after a context-aware system processes its inputs.

This paper is in favour of the 'context as a derivable' interpretation of context. This interpretation implies that context is what a context-aware system is programmed to accomplish. This interpretation also implies that context, as is used by the majority of the researchers, is a context parameter. It is an input that is essential for a context-aware system to accomplish its tasks. This *programmed* context is static and hence runs counter to the reasons for inventing context-aware systems in the first place. It is possible to develop context-aware systems that utilise data from numerous sensors and other sources to determine and subsequently to adapt to their environments. This requires huge processing power, which not all hosting computing devices have. Additionally, the form factor of the majority of hosting devices is not flexible to accommodate new sensors as are invented. Hence, we argue that architectural solutions are required in order to accommodate the adaptive nature of context-aware systems.

We have briefly described our Knowledge-driven Distributed Architecture (KoDA) which despite of monitoring and interpreting information from sensors, it reasons about the available information to recognise ongoing context. Among the benefits of KoDA includes the simplification of the process of implementing context-aware systems and the removal of redundancy of inference rules. In KoDA, inference rules are not represented in context-aware systems. This significantly simplified the process of implementing context-aware systems as developers are freed from the hurdles of learning knowledge representation languages required to represent inference rules. In KoDA, inference rules are centralised and hence removes redundancy of inference rules.

CONCLUSION

[1] Schilit, B., Adams, N. & Want, R. (1994). Context-aware computing applications. In *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*, 85 - 90.

[2] Dey, A. & Abowd, G. (2000a). Towards a better understanding of context and context-awareness. In *CHI 2000 workshop on the what, who, where, when, and how of context-awareness*, 304 – 307.

[3] Zimmermann, A., Lorenz, A. & Oppermann, R. (2007). An operational definition of context. In *Modeling and using context*, 558 - 571, Springer.

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

[5] Want, R., Hopper, A., Falc~ao, V. & Gibbons, J. (1992). The active badge location system. *ACM Trans. Inf. Syst.*, 10, 91 - 102.

[6] Schilit, B. & Theimer, M. (1994). Disseminating active map information to mobile hosts. *Network, IEEE*, 8, 22 - 32.

[7] Pascoe, J. (1998). Adding generic contextual capabilities to wearable computers. In *Wearable Computers, 1998. Digest of Papers. Second International Symposium on*, 92 - 99, IEEE.

[8] Ryan, N.S., Pascoe, J. & Morse, D.R. (1998). Enhanced reality fieldwork: the context-aware archaeological assistant. In *Computer applications in archaeology*, Tempus Reparatum.

[9] Abowd, G., Atkeson, C., Hong, J., Long, S., Kooper, R. & Pinkerton, M. (1997). Cyberguide: A mobile contextaware tour guide. *Wireless Networks*, 3, 421 - 433.

[10] Weiser, M. (1991). The computer for the 21st century. *Scientific American*.

[11] Schilit, B.N., LaMarca, A., Borriello, G., Griswold, W.G., McDonald, D., Lazowska, E., Balachandran, A., Hong, J. & Iver-son, V. (2003). Challenge: Ubiquitous location-aware computing and the place lab initiative. In *Proceedings of the 1st ACM international workshop on Wireless mobile applications and services on WLAN hotspots*, 29 - 35, ACM.

[12] Soylu, A., Causmaecker, P.D. & Desmet, P. (2009). Context and adaptivity in pervasive computing environments: Links with software engineering and ontological engineering. *Journal of Software*, 4, 992 - 1013.

[13] Liu, H. (2010). Biosignal controlled recommendation in entertainment systems. *Technische Universiteit Eindhoven, Eindhoven*, 1 - 133.

[14] van de Westelaken, R., Hu, J., Liu, H. & Rauterberg, M. (2011). Embedding gesture recognition into airplane seats for in-ight entertainment. *Journal of Ambient Intelligence and Humanized Computing*, 2, 103 - 112.

[15] Chen, G. & Kotz, D. (2000). A survey of context-aware mobile computing research. Tech. rep., Citeseer.

[16] Chen, H. (2004). *An Intelligent Broker Architecture for Pervasive Context-Aware Systems*. Ph.D. thesis, University of Maryland.

[17] Kaenampornpan, M. (2009). *A Context Model, Design Tool and Architecture for Context-Aware Systems Design*. Ph.D. thesis, Department of Computer Science, University of Bath.

[18] Kofod-Petersen, A. (2007). *A Case-Based Approach to Realising Ambient Intelligence among Agents*. Ph.D. thesis, Department of omputer and Information Science, Norwegian University of Science and Technology.

doi:10.15849/icit.2015.0111 © ICIT 2015 (http://icit.zuj.edu.jo/ICIT15)

[19] Bolchini, C., Curino, C.A., Quintarelli, E., Schreiber, F.A. & Tanca, L. (2007). A data-oriented survey of context models. SIGMOD Rec., 36, 19 - 26.

[20] Gellersen, H.W., Schmidt, A. & Beigl, M. (2002). Multi-sensor contextawareness in mobile devices and smart artifacts. *Mobile Networks and Applications*, 7, 341 - 351.

[21] Barkhuus, L. (2005). *The Context Gap: An Essential Challenge to Context-Aware Computing*. Ph.D. thesis, The IT University of Copenhagen.

[22] McKeever, S. (2011). *Recognising Situations Using Extended Dempster-Shafer Theory*. Ph.D. thesis, School of Computer Science and Informatics, National University of Ireland.

[23] Lupiana, D. (2015). *A Knowledge-driven Distributed Architecture for Context-Aware Systems*. Ph.D. thesis, School of Computing, Dublin Institute of Technology.

[24] Gu, T., Pung, H.K. & Zhang, D.Q. (2005). A service-oriented middleware for building context-aware services. *Journal of Network and computer applications*, 28, 1 - 18.

[25] Ye, J., Coyle, L., Dobson, S. & Nixon, P. (2007). Using situation lattices to model and reason about context. In *Modeling and Reasoning in Context (MRC) with Special Session on the Role of Contextualization in Human Tasks (CHUT) which is held in conjunction with CONTEXT*, 1 - 12.

[26] Henricksen, K. (2003). *A Framework for Context-Aware Pervasive Computing Applications*. Ph.D. thesis, School of Information Technology and Electrical Engineering, The University of Queensland.

[27] Ranganathan, A. & Campbell, R.H. (2003a). *An infrastructure for context-awareness based on first order logic. Personal and Ubiquitous Computing*, 7, 353 - 364.