International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 6, No.5 (Sep-2017)
http://dx.doi.org/10.12785/ijcds/060506

A Survey of Adaptation and the Best Approach for Ubiquitous Systems

Abdulgader Almutairi[1], and Mafawez Alharbi [2]

[1] *Computer Science department, College of Art &Science, Qassim University, Saudi Arabia*
[2] *Computer Science Department, College of Science, Majmaah University, Saudi Arabia*

Received 7 Jul. 2017, Revised 18 Aug. 2017, Accepted 23 Aug. 2017, Published 1 Sep. 2017

Abstract: Adaptation is defined as the process of changing to suit different conditions in the environment. In other words, adaptation is a change in the structure, function or behaviour of the system in order to meet the current context. A ubiquitous system refers to the manner in which computer systems adapt to changes in the physical and virtual environments of computing platforms and communication networks, whilst enabling better interaction between humans and computers. The development and implementation of adaptive systems is rooted in the study of evolution and adaptation in the natural environment; accordingly, new methods have been continuously developed to enable computational adaptation. Adaptive computing is thus generally focused on the rapidly growing areas of, and the increasing demands for, ubiquitous computing and autonomic computing. This study first describes the notion of adaptation, before explaining adaptive systems and discussing adaptation in ubiquitous computing. The adaptation process in adaptive systems that occur in runtime stages is then presented. Moreover, the classification of adaptation is discussed in terms of questions relating to who, why and what, as well as how the adaptation occurs. Finally, different adaptation approaches are demonstrated and compared in order to select the most suitable approach for ubiquitous systems.

Keywords: Adaptive system; Adaptation Process; Ubiquitous System.

1. INTRODUCTION

Adaptive computing has increased dramatically in recent years and has become one of the most dominant research areas [21]. Computer systems use a variety of techniques to enable software to adapt to existing use and environmental changes [10]. The software structure may be changed during performance in order to correct errors, increase the effectiveness of its procedures, increase fault tolerance, and strengthen computer security .The development and implementation of adaptive systems is rooted in the study of evolution and adaptation in the natural environment; new methods have accordingly been continuously developed to enable computational adaptation. Adaptive computing is thus generally focused on the rapidly growing areas of, and the increasing demands for, ubiquitous computing and autonomic computing [1].

Ubiquitous computing refers to the manner in which computer systems adapt to changes in the physical and virtual environments of computing platforms and communication networks, whilst enabling better interaction between humans and computers [14, 4]. Computer software needs to confront and adjust to numerous highly dynamic obstacles, such as energy consumption, changing security policies, and quality-of-service in wireless connections [12, 8].

Autonomic computing relates to how a system can manage and configure itself to achieve high-level goals. It is thus the case that autonomic computing is an important factor in ubiquitous computing [7] [5]. An autonomic computing system has the ability to manage and protect its own resources; this is highly desirable in systems that must continue to operate during exceptional circumstances, such as in transportation systems or finance networks. This requires continues adaptation in the event of hardware component failure, network outages and security attacks [6].

Therefore, most recent research has aimed to develop the most appropriate adaptation approaches that cover the characteristics of ubiquitous systems. A limited number of studies have examined the approaches to adaptation used in ubiquitous systems. However, this research has not produced optimal approaches and more improvement is needed in order to obtain a suitable role

E-mail: Azmtierie@qu.edu.sa, M.alharbi@mu.edu.sa

model [13, 17, 18, and 23]. This study therefore aimed to compare the different adaptation approaches presented in this paper in order to highlight the most suitable approaches for ubiquitous systems.

This paper presents a taxonomy of adaptation. It is organized as follows: Section 2 presents a brief background and related work on adaptation approaches. Section 3 gives an overview of adaptive systems. Section 4 describes the adaptation process. Adaptation in ubiquitous systems is presented in Section 5. The taxonomy of adaptation is discussed in Section 6. The results and findings are given in Section 7. Finally, the paper summarized in Section 8.

2. RELATED WORK

The literature shows that various adaptation approaches have been utilized in order to realize dynamic adaptation in software. In the following subsections, three adaptation approaches are presented, known as parameter adaptation, compositional adaptation and action-based adaptation [3].

A. Parameter Adaptation

This kind of adaptation approach is utilized to affect the behaviour of the system via the amendment of particular variables in the program. A common example of this approach is the TCP protocol, in which its behaviour can be adjusted according to network congestion. Parameter adaptation is used in the area of ubiquitous computing to amend the non-functional properties of a system that are influenced by context related change. For instance, a portable device that has an image render might present low-quality images because of the low bandwidth or low memory size. The system thus adapts its behaviour to the current situation by adjusting particular parameters in the image application in order to meet the requirements of the new situation.

However, having unpredictable context changes in a highly dynamic environment makes it difficult to try to define in advance all the probable contexts (as well as their processes) for non-functional properties. The advantage of the parameter adaptation approach is that it is cheap in terms of implementation effort and complexity, which is why some context-aware systems use this approach [6].

On the other hand, the parameter adaptation approach is considered an optimal solution in the area of ubiquitous systems; its main disadvantage is that the software components (as well as unimplemented algorithms) that are left throughout the design stage cannot be adapted. In addition, using this approach for such an application, where its behaviour is based on frequent context changes, might lead to several configurable parameters becoming conflicted. Thus, it is very important to adopt a different approach, namely one that helps to reduce the number of parameters.

B. Compositional Adaptation

Another approach to adaptation is a compositional adaptation; this goes further than straightforward code-tuning and permits the algorithm or parts of the system structure to be replaced to enhance the program in order to meet the current situation. Compositional adaptation works in environments with unpredictable contexts or requirements and where new adaptation functionality may be required. The most important technologies used in compositional adaptation are thus discussed below:

- Separation of Concerns: in this technology, the segregation of functional from non-functional behaviour in terms of development is permitted if the functional behaviour is related to business logic and the non-functional behaviour is associated with the security and quality of service. This mechanism facilitates the development of the system and its maintenance when upgrading the system. Separation of concerns, such as domain-specific languages, constraint languages and generic languages, has been considered a significant principle in the area of software engineering, and is employed in various advanced techniques.

- Computational Reflection: this is related to the ability of an application to process new contexts and adapt its behaviour to the current situation. It facilitates the changing of a system’s behaviour through expressing the implementation details of the system at an abstract level, devoid of negotiating portability. In this technology, two activities are introduced: introspection and intercession; the former deals with the observation of the system behaviour, and the latter responds to the changes that are captured in the observation stage and then adapt to the new situation.

C. Action-based adaptation

This kind of adaptation approach is also known as rule-based adaptation. Action-based adaptation is a common approach employed in adaptive systems. It is able to define the self-configuring and self-managing aspects of the system's behaviour that are associated with distributive technologies and networks. This approach relies on the concepts of state and action, where the system at a specific time (t) should move from the current state (S1) to the next state (S2) if all corresponding conditions are true and the transition of the system has to be determined by the action (a). Therefore, the system is directed to adapt to the new situation through using the format of If (conditions), and Then (actions).

Many researchers have used this type of adaptation approach in their work. One common model proposed [24], developed an adaptation platform for mobile systems; it uses action policies based on an event calculus, which are formulated in the form of conditions and actions that accurately determine the adaptation behaviour of the system. The conditions are defined as logical expressions that may take the value 'true' or 'false', while the actions are defined as the adaptation methods that are performed if the condition is evaluated to be 'true'.

Moreover, the notion of event-action rules has been used in terms of expressing dynamic system reconfiguration. For instance, the DART project [13] uses an adaptation manger that implements adaptation policies, which are activated by particular events created depending on user requirements and system statistics. Thus, each policy is associated with one or more events, and whenever a specific event occurs, the most suitable policies are invoked by the manager, who then executes them. The problem is that many policies might index the same event, which could cause the policies to conflict; in order to solve this problem, appropriate priorities are allocated to each policy.

3. ADAPTIVE SYSTEM

There are a variety of definitions of adaptive systems, one of the most common of which is that of [9], which states, "Self-adaptive software evaluates its own behaviour and changes behaviour when the evaluation indicates that it is not accomplishing what the software is intended to do, or when better functionality or performance is possible." The most appropriate definition of a ubiquitous system was given by [15], who stated that, "Self-adaptive software modifies its own behaviour in response to changes in its operating environment. By operating environment, we mean anything observable by the software system, such as end-user input, external hardware devices and sensors, or program instrumentation" [13]. The abstract architecture of adaptive systems can be seen in Fig1.

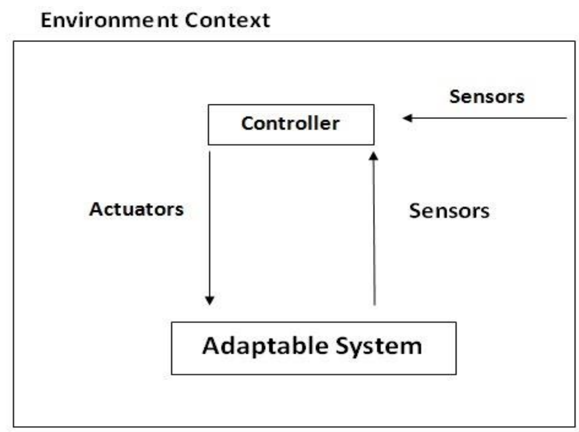


Figure 1: Adaptive system architecture.

There are different kinds of systems that have linked and similar meanings to the term adaptive system, including self-managing and autonomic systems. However, it is difficult to identify any real differences between these terms, and they are used interchangeably by the majority of researchers. Nevertheless, the distinction an between autonomic and self-adaptive system is that the former is more general, whereas the latter is more restricted, which means that an adaptive system is a special case of an autonomic system [19].

4. ADAPTATION PROCESSES

This section presents the adaptation processes that take place during the run-time stages [2], as shown in Fig 2.

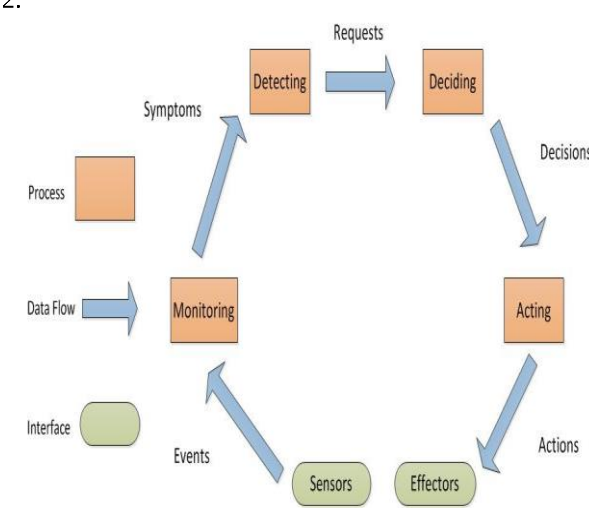


Figure 2. The Four Adaptation Processes in Adaptive Systems.

The first stage in adaptation is the monitoring process, which involves gathering all the data that are observed by sensors in order to transfer them into behavioural symptoms and models. In this process, all the different questions that occur in the run-time stage are identified, such as where, when and what questions. The realization of this process can be achieved through various means, such as threshold checking and occasion correlation.

The second stage in adaptation is the detecting process, the main role of which is to analyse the symptoms received from the first stage along with the system's history in order to determine a suitable time for the response. Another role of this process is to identify the new destination or state.

The third stage in adaptation is the deciding process, which defines the change that must occur and the most suitable way to effect this change in order to accomplish a satisfactory result. This must be based on certain criteria in order to compare the possible changes in the approaches, such as choosing between different sets of actions.

The final stage in adaptation is the acting process, the main role of which is to perform the special and particular actions that were decided upon by the previous stage, and to manage non-primitive actions via predetermined workflows. The acting process is associated with various questions such as how, when and what to amend.

5. ADAPTATION IN UBIQUITOUS COMPUTING

Ubiquitous computing adaptation is considered a reactive process in which the adaptation takes place based on a particular circumstance or series of events in the environmental context; the main objective of this adaptation is to improve the quality of the service used by the system user. Therefore, the most significant requirements when an application is to be used in the ubiquitous environment are an ability to sense the surrounding environment and process these changes, and to respond to these changes in an efficient and effective manner [1].

The most common description for adaptation within ubiquitous computing is taken from the MAPE-K loop developed by IBM. This is often used in the autonomic computing context. Thus, adaptation in ubiquitous computing is considered a closed loop consisting of different phases [11], as shown in Fig3.

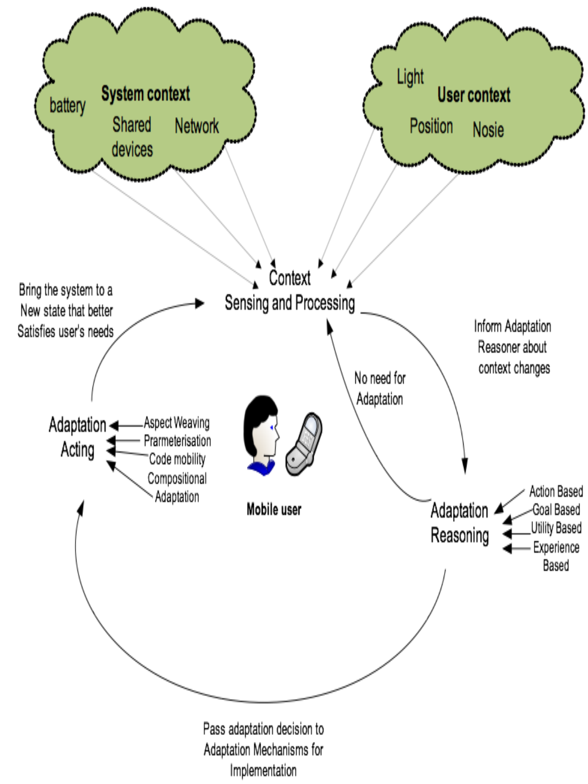


Figure 3. MAPE-K loop; IBM.

The first phase consists of the sensing and processing of context; this phase senses the different user contexts, such as location and user preferences, and the different system contexts, such as light level and temperature. All the gathered data are interpreted in terms of high-level context events in order to consider the various available steps in the adaptation process steps.

The next phase is reasoning and planning; the function of this phase is to process and analyse any new changes to the context that have been captured by the sensors, and to consider the particular type of adaptation required as well as how to accomplish the required objectives.

The final phase in ubiquitous adaptation is adaptation acting; this phase is responsible for implementing the most suitable adaptation approach in order to affect the adaptation decisions in the previous phase (reasoning and planning process).

6. ADAPTAION TXONOMY

In this section, a graphical illustration of the adaptation taxonomy is presented (in Fig4). This also classifies the adaptation methods using questions relating to who, why, what, and how the adaptation is to take place [22,16, 23].

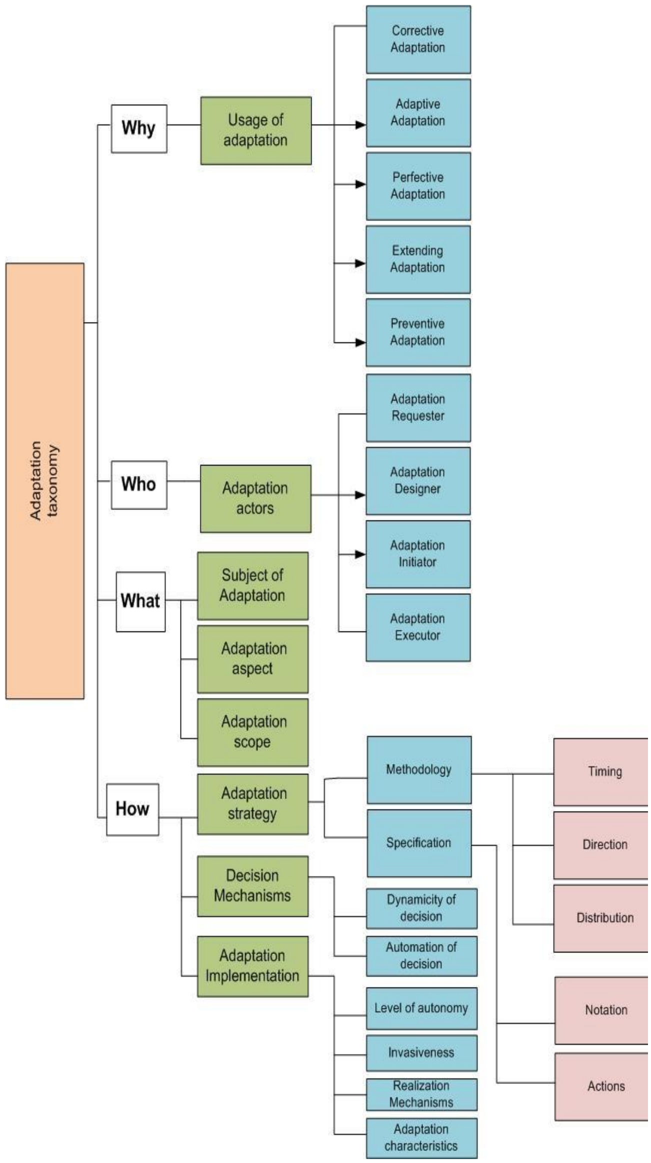


Figure 4. Adaptation taxonomy [23].

A. The Why Dimension

The four dimensions of the adaptation classification determine the most suitable adaptation approach. This dimension offers an explanation for the reason behind initiating an adaptation [20]. The required adaptation process can be identified through its objective, which can be categorized as follows:

- Perfective Adaptation: in this case, the adaptation takes place in terms of enhancing the system functionalities, such as improving the service quality, although the system may already be working properly.
- Corrective Adaptation: this type of adaptation is used to eliminate defective performance, changing the system to an updated version that performs the same functions. Many different types of faults occur in decentralized systems; handling such failures is important in order to ensure that the system behaves in the desired manner.
- Adaptive Adaptation: this type of adaptation is used to cope with dynamic changes in the surrounding environment; the system should adapt itself based on the context of its environment in order to improve the quality of service.
- Preventive Adaptation: in this case, the adaptation is used in order to protect the system from possible failures.
- Extending Adaptation: this kind of adaptation is used to expand the scope of the system tasks via introducing new functions as required.

These adaptation classifications can be further categorized based on special issues. For instance, context-aware, customization and mediation adaptation can be inherited from adaptive adaptation.

B. The Who Dimension

The next dimension is the Who dimension, which describes the problem of adaptation based on the perspectives of the various actors (human and software) that are involved in the process of adaptation:

- Adaptation Requestor: this represents the stakeholder who determines the requirements for the adaptation of the system.
- Adaptation Designer: this actor is responsible for determining the adaptation plan in order to accomplish the adaptation requirements.
- Adaptation Initiator: this is responsible for commencing the system amendments in response to recognized changes.
- Adaptation Executor: this actor is responsible for implementing the adaptation actions that are determined by the adaptation strategy.

C. The What Dimension

In this dimension, the adaptation aims and objectives are classified, and the subsequent elements of this dimension are considered:

- Subject of Adaptation: this element identifies the entity that has to be adapted via the process of adaptation.
- Adaptation Aspect: this element considers the special concerns of the adaptation process, such as the quality of the system model (usability, security).
- Adaptation Scope: this element measures the influence of the adaptation process to determine whether the adaptation will be for the short or the long term.

D. The How Dimension

The last dimension of the adaptation taxonomy is the how dimension, which is defined as the means with which to apply and accomplish the adaptation; this can be done by specifying the particular strategic approaches, decision mechanisms and implementation approaches.

- *Adaptation Strategies*

This is defined as identifying the approach that would best meet the requirements and aims of the adaptation, such as re-binding, re-execution and re-configuration. The classification of adaptation strategies is based on various characteristics, such as the location of adaptation, the methodology that is to be employed, and the means for specifying the strategy.

Adaptation Methodology: this consists of the timing, the distribution and the adaptation direction, as follows.

- Timing is defined as the actual time and duration of the adaptation process. There are two types of adaptation, known as reactive adaptation and proactive adaptation; the former relates to amendments in response to the changes that previously took place, while the latter relates to the adaptation that will take place before the changes actually occur during the actual system operation.
- Direction of adaptation: there are two directions of adaptations, defined as forward adaptation and backward adaptation; forward adaptation uses the adaptation strategy in order to transit the system from its current state to the new 'forward' state to fulfil the adaptation requirements, while backward adaptation reverts the system to the previous state that has already fulfilled the adaptation requirements.
- Distribution of the adaptation: this differentiates between two types of adaptation: centralised adaptation and decentralised adaptation. The former presents the adaptation process occurring centrally, namely on the server side only, while the latter presents the distributed adaptation process, namely on the client side.

Adaptation Specification: this illustrates the notations used for the strategies and actions, demonstrating those strategies in order to specify them. These notations can be implicit; in this case, the adaptation strategies and actions are represented as hard-coded inside the system, based on a predefined plan, and cannot be amended unless the adaptation mechanism is changed.

- Action-based specification: in this case, the particular actions are specified to be performed in certain conditions or situations.
- Goal-based specification: this type of specification is responsible for launching objective performance and for enabling the system to define the particular actions that lead to accomplishment of the objectives.

Adaptation action: this represents an action that has been executed in the system in order to adapt to the new situation, based on the current context and to satisfy the adaptation requirements. It determines the adaptation strategy semantics. The classification of the adaptation actions can be presented based on the subject of the adaptation process and its scope, such as retry, substitute service, skip, undo, or choose alternative behaviour.

- *Decision Mechanisms*

These types of mechanisms are responsible for choosing the most suitable adaptation method for implementing the strategy decision in order to fulfil the adaptation requirements. These mechanisms can be considered in terms of strategy selection: dynamicity and automation.

- Dynamicity of decision: this is related to flexibility in choosing the adaptation method to fulfill the strategy. The selection of this decision can be classified into three types of selections, the first of which, static selection, means that the predefined adaptation strategy can be associated with a specified adaptation requirement or condition. The second selection is dynamic selection, which means that the selection of an adaptation strategy will be at run-time and will be based on the current context, information and situation. The last selection is evolution-based, which means that the selection of the adaptation strategy is based on the present context and as well as all the preceding adaptation, decision and related consequences.

- Automation of decision: this mechanism describes the level of participation of humans in the decision process. This can be entirely automatic, which means no user involvement in decision process or the user selecting as an intervention in the decision process.

- *Adaptation Implementation*

Adaptation Implementation: this represents the way in which the methodology and adaptation architecture can be realized. It is determined using several mechanisms, such as autonomy implementation, framework invasiveness, realization mechanism and particular characteristics, which permit the approach to be measured.

- Autonomy describes human intervention in the execution of adaptation. It can be performed without human intervention (namely autonomously) or with user involvement in the adaptation execution.

- Invasiveness illustrates the adaptation framework based on a strong integration between the framework execution and the adaptation subject. It can distinguish between various cases; first, the integration of adaptation facilities with the subject; secondly, the integration of adaptation facilities with the platform; and lastly, the segregation of adaptation facilities from the adaptation subject.

- Realization mechanisms describe the appropriate facilities and tools that are required to allow the adaptation methodology to apply the adaptation strategy, in order to create the adaptation architecture. This mechanism is based on the specified adaptation problem and the method that is utilized for it.

Adaptation characteristics: these explain various issues that should be satisfied by the adaptation process, such as security and safety, as well as the cost of the adaptation process.

7. RESULTS AND FINDINGS

This section presents the differences between all the adaptation approaches mentioned above in order to identify the one best suited to the field of ubiquitous computing. The parameter adaptation approach is suitable for a low-dynamic environment; it can change the system's behaviour by alerting particular parameters that are influenced by context changes. However, this adaptation approach is not appropriate for highly dynamic environments, which must consider all the various types of context that could occur and respond based on those contextual changes. The second adaptation approach is compositional; this is considered a general approach to self-adaptation based on the language of implementation. Based on several studies that have been accomplished through compositional adaptation, it may be argued that compositional adaptation is a powerful approach for two different reasons; the first, in terms of processing level, is its flexibility of reasoning, and the second, in terms of adaptation reaction, is the easy implementation of the dynamic reconfiguration of the system (compositional adaptation allows algorithmic and structural changes).

However, the action-based adaptation approach is considered a good alternative solution; it uses actions in order to adapt to new situations. This type of adaptation approach is more suitable for non-functional requirements, such as security and quality of service. Based on adaptation mechanisms in ubiquitous systems, this type of adaptation approach is preferred; as such systems use adaptation actions to adapt to the current context. The action-based adaptation approach uses conditions and actions; the actions take place if the conditions are evaluated as being false in the adaptation process.

8. CONCLUSION

In this paper, an overview and background of the taxonomy of adaptation was provided. The background and definitions of adaptive systems in general were also described. Subsequently, adaptation in ubiquitous systems, the adaptation process, and various aspects of adaptation in terms of adaptation approaches, were also discussed. Different adaptation approaches were investigated and compared. The results and findings section shows that the action-based adaptation approach is considered a good alternative solution. It is hoped that this systematic survey and taxonomy will inspire more research into intelligent ubiquitous systems and their interaction with other problems in the ubiquitous system.

REFERENCES

[1] K. Kakousis, N. Paspallis, and G. A. Papadopoulos, "A survey of software adaptation in mobile and ubiquitous computing,"Enterp. Inf. Syst., 4(4),pp.355–389, Nov. 2010.

[2] M. Rohr, S. Giesecke, M. Hiel, W.-J.van den Heuvel, H.Weigand, and W. Hasselbring, "A classification scheme for self-adaptation research," In Proceedings of the International Conference on Self-Organization and Autonomous Systems In Computing and Communications (SOAS'2006), pp. 5, 2006.

[3] S.-H. Chang, H. J. La, and S. D. Kim, "A comprehensive approach to service Adaptation," In Service-Oriented Computing and Applications, 2007. SOCA '07. IEEE International Conference on, pp.191–198, 2007.

[4] A. K. Dey, "Understanding and using context," Personal Ubiquitous Comput., 5(1), pp. 4–7, Jan. 2001.

[5] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli, "A survey of autonomic Communications," ACM Trans. Auton. Adapt. Syst., 1(2), pp.223–259, Dec. 2006.

[6] J. Fox and S. Clarke, "Exploring approaches to dynamic adaptation,"In Proceedings of the 3rd International DiscCoTec Workshop on Middleware-Application Interaction, MAI '09, pp.19–24, New York, NY, USA, 2009. ACM.

[7] M. Friedewald and O. Raabe, "Ubiquitous computing: An overview of technology Impacts," Telematics and Informatics, 28(2),pp.55 – 65, 2011.

[8] R. Hamadi and B. Benatallah, "Recovery nets: Towards self-adaptive workflow Systems,"In in Proceedings of the 5th International Conference on Web Information Systems Engineering (WISE 04), LNCS 3306, pp. 439–453. Springer Verlag, 2004.

[9] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," ACM Trans. Auton. Adapt. Syst., vol. 4, no. 2, pp. 14:1–14:42, May 2009. [Online]. Available: http: //doi.acm.org/10.1145/1516533.1516538

[10] C. Klein, R. Schmid, C. Leuxner, W. Sitou, and B. Spanfelner, "A survey of context adaptation in autonomic computing," In Proceedings of the Fourth International Conference on Autonomic and Autonomous Systems, ICAS '08, pp. 106–111, Washington, DC, USA, 2008. IEEE Computer Society.

[11] P. Lefrere, "Activity-based scenarios for and approaches to ubiquitous elearning,"Personal Ubiquitous Comput., 13(3):pp. 219–227, Mar. 2009.

[12] S. LOKE, "Context-Aware Pervasive system: Architectures for new breed of Applications," Auerbach Publications, Wiley, 2006.

[13] P. Oreizy, N. Medvidovic, and R. N. Taylor, "Runtime software adaptation: framework, approaches, and styles," In Companion of the 30th international conference on Software engineering, ICSE Companion '08, New York, NY, USA, 2008.ACM.

[14] S. Poslad, "Ubiquitous Computing: Smart Devices, Environments and Interactions," Auerbach Publications, Wiley, 2009.

[15] M. Rohr, S. Giesecke, W. Hasselbring, M. Hiel, W.-J.van den Heuvel, and H. Weigand, "A classification scheme for self-adaptation research," In Proceedings of the International Conference on Self-Organization and Autonomous Systems In Computing and Communications (SOAS'2006), September 2006.

[16] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research Challenges," ACM Trans. Auton. Adapt. Syst., 4(2):pp. 14:1–14:42, May 2009.

[17] X. Zhao, X. Wan, and T. Okamoto, "Adaptive content delivery in ubiquitous learning environment," In Wireless, Mobile and Ubiquitous Technologies in Education (WMUTE), 2010 6th IEEE International Conference on, pp. 19–26, 2010.

[18] C. Bolchini, M. Carminati, A. Miele, and E. Quintarelli, "A framework to model self-adaptive computing systems," in Adaptive Hardware andSystems (AHS), 2013 NASA/ESA Conference on, June 2013, pp. 71–78.

[19] T. Chen and R. Bahsoon, "Self-adaptive and online qos modeling for cloud-based software services," IEEE Transactions on SoftwareEngineering, 2016.

[20] J. Kramer and J. Magee, "Self-managed systems: An architectural challenge," in FOSE, 2007.

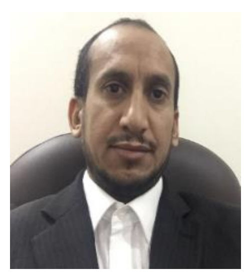
[21] R. de Lemos et al., "Software engineering for self-adaptive systems: A second research roadmap," ser. LNCS, vol. 7475. Springer, 2013.

[22] D. Weyns and T. Ahmad, "Claims and evidence for architecture-based self-adaptation: A systematic literature review," in Software Architecture, ser. Lecture Notes in Computer Science, 2013, vol. 7957.

[23] L. Tang, H. Liu, J. Zhang, N. Agarwal, and J. J. Salerno, "Topic taxonomy adaptation for group profiling," ACM Trans. Knowl. Discov. Data, 1(4):1:1–1:28, Feb. 2008.

[24] C. Efstratiou, A. Friday, N. Davies, and K. Cheverst,"Utilising the event calculus for policy driven adaptation on mobile systems," In Policies for Distributed Systems and Networks, 2002. Proceedings. Third International Workshop on, pp.13–24, 2002.

Abdulgader Almutairi was born in 1980 at alQassim, Saudi Arabia. He obtained his Bachelor degree in 2003 in Computer Science at King Saud University in Saudi Arabia. Then he received his Master degree in 2009 in Software Engineering at De Montfort University in UK, and he gained from the same university his PHD degree in 2013 in Computer Science. Dr. Almutairi is currently an Assistant Professor at Al Qassim University and he is also a consultant at IT Unit in Arass College. He has more than five published papers in international conferences. His research interests include context-aware system, access control, cloud computing, usage control.



Mafawez Alharbi is an assistant professor of computer science department, vice dean of IT deanship in Majmaah University and academic consultant, Saudi Arabia. He received a bachelor's degree in computer science in 2004 from King Saud University, Saudi Arabia, a master's degree in software engineering in 2010 from De Montfort University, and his PhD in computer Science in 2014 from De Montfort University, Leicester, United Kingdom. His PhD thesis was entitled as title is "Context-aware Personal Learning Environment"; Mafawez has published several scientific Papers in international journal and conferences.