An SOA based ubiquitous computing system design framework

IDOUGHI Djilali
Applied Mathematics Laboratory - LMA
University A. Mira of Bejaia
Bejaia, Algeria
djilali.idoughi@gmail.com

AZOUI Aicha
Applied Mathematics Laboratory - LMA
University A. Mira of Bejaia
Bejaia, Algeria
aicha.azoui@gmail.com

ABSTRACT
Ubiquitous services are software applications that have the capability to run anytime, anywhere and on any device with minimal or no user attention. However, the advancements and diversity in mobile technologies, the dynamic and ubiquitous nature of these services increase the complexity of the development process. In this paper, we propose an approach for the design and development of service oriented ubiquitous systems considering both business and ubiquitous requirements. The approach is then applied to a crisis management case study.

Keywords
Ubiquitous systems; service orientation; ubiquitous service; ubiquitous requirements; design framework.

1. INTRODUCTION
Ubiquitous computing is the third wave of computing technologies to emerge since computers first appeared [1]. The term as defined by Mark Weiser [2] is currently used to describe a computing environment in which computation is everywhere and computer functions are embedded and connected with all things so that anyone can access, communicate and share information anywhere and anytime. The figure 1 illustrates broadly the current vision of ubiquitous computing environment. This illustration shows the environment which surrounds the user’s everyday lives and activities. This environment, presented as circles around users, is saturated with computing devices and communication capabilities, which can be implemented in a varied scale of spaces where information and services are provided to the users when and where desired. The convergence of mobile technologies in terms of mobile devices and telecommunication and software engineering paradigms (especially the emergence of service oriented paradigm) has brought about a new service-oriented computing paradigm known as ubiquitous services or context-aware services [3]. However, these advancements and diversity in technologies and the highly dynamic nature of ubiquitous services contribute to make ubiquitous services design complex and challenging compared to conventional services. Thus, the design and the development of ubiquitous services is a more complex task [3][5] than in a conventional service design approaches where input information can be managed in a common way since it is mainly considered to be obtained principally from the user. In contrast, when developing ubiquitous services, this needs to be managed differently since it is obtained from diverse sources [6].

In this paper, we aim to present an approach to developing ubiquitous systems based on the service-oriented paradigm providing the end-user with ubiquitous services meeting their needs, and that adapts to their context. The Remainder of this paper is organized as follows: section 2 presents the service oriented computing concept. Section 3 introduces ubiquitous services, and a definition is given. Section 4 presents some related works on the development of service oriented ubiquitous systems along with their drawbacks and limitations. Section 5 presents the basic concepts and foundations for our approach. The approach framework is described in section 6. And finally, we conclude and give some further research work in the section 7.

2. SERVICE ORIENTED COMPUTING (SOC)
Service-Oriented Computing (SOC) is the computing paradigm that utilizes services as fundamental elements for developing applications [7]. In SOC, a service is defined as a software resource (discoverable) with an externalized service description. This service description is available for searching, binding, and invocation by a service consumer [8]. Specifically, an application is built as a composition of a set of services, where each service acts as a fundamental building block. Reusability, simplicity and interoperability are some design objectives of those services. The services are characterized to be: loosely coupled, distributed, invocable, publishable and business oriented [16]. Service-Oriented Architecture (SOA) is a supporting infrastructure providing all the features necessary to implement SOC in a given technological domain.

3. UBIQUITOUS SERVICES
Ubiquitous services are aware of the dynamic environment in which they evolve, compared to traditional services which are poorly informed about their surroundings. In a ubiquitous environment, ubiquitous services can interact with several users,
each with specific needs and who operate in different spaces populated by different technologies. So, a ubiquitous service must take into account the particular context of each user invoking it in his environment in order to satisfy his/her needs. However few definitions are given to ubiquitous services in the literature. According to [12], ubiquitous services are software applications that can operate in a dynamic environment and have the capability to run anytime, anywhere and on any device with limited or no user attention.

4. RELATED WORK
The development of Ubicomp services based system has led to the emergence of different works that provide frameworks and architectures for this type of systems. However, the most proposed works concentrate only in the treatment of one aspect related to the service such as discovery [15], composition [11] or adaptation [14]. Chaari et al. [14] focus on the adaptation of services to different context situations in a ubiquitous environment. Toninelli et al. [15], in a ubiquitous scenario, provide a middleware which adopts semantic techniques to perform the discovery of context aware services based on the requirements and preferences expressed by mobile users. Tigli et al. [11] propose an architecture model for the service composition which is based on an assembly of lightweight components. Moreover, some other research work [6], [17], [18], [19] suggest the employment of a model driven engineering process [20] which is an approach for using models at various levels of abstraction in software development. The key idea is to automatically transform highly abstract models into more concrete models from which an implementation can be generated in a straightforward way. Therefore, Achilleos et al. [6] propose a model-driven development process that facilitates the creation of a context modeling framework and simplifies the design and implementation of ubiquitous services. Vale and Hammoudi [17] propose the use of the model driven engineering approach to develop context-aware services by the separation of concerns (i.e., business, context, context awareness) in different models. However, analysis requirements is not taken into consideration, as well as the proposed context meta-model is not generic and the adaptation strategy is not described. Sheng and Benatallah in [18] present a modeling language based on UML for the model driven development of context aware services and a generic meta-model of context is presented. The proposed approach does not provide guides and instructions for context modeling and requirements analysis. Moreover, the authors do not specify the mechanism used to fulfill context aware services adaptation. Finally, Abeywickrama, and Ramakrishnan [19] propose an engineering approach for context aware services that models and verifies these services at the architectural level. The approach benefits of several software engineering principles such as the model driven architecture, the separation of concerns through aspect-oriented modeling and formal verification using model checking to facilitate context aware services engineering. The approach development process is incomplete because it does not take into account the analysis phase and a context model is not given. Most approaches cited above have focused on the context modeling and adaptation to context changing as dominant and unique aspects of ubiquitous systems. However, the functional aspect of the system is neglected which makes these approaches incomplete. In addition, the requirements analysis phase identifying business and ubiquitous requirements yielding to necessary services and model the context is absent in all the approaches presented. Therefore, the aim of this paper is to present an approach for the development of Ubicomp service based systems which overcomes these lacks and provides a richer engineering process.

5. THE APPROACH BASIC CONCEPTS
The proposed approach is based on a set of principles considered as foundations facilitating separation of the developer’s tasks and specification of ubiquitous models. It considers ubiquitous services as a central concept resulting from building SOA business and ubiquitous view. Thus, we separate the development process into business and ubiquitous views. With such separation, the software is not addressed as a whole, but in parts. The business view focuses on the business logic of the system and its functionalities, independently of ubiquitous and technological aspects. The ubiquitous view is responsible for the definition of context data, their acquisition, interpretation, modeling and their exploitation by the application. This separation is due to the reasons that the contextual data that form the basis of all the ubiquitous requirements are different from system data and do not affect the business and the system functionalities. They generally represent parameters added to the business to determine how these functionalities are realized. We may distinguish four categories of actors involved in the development process of ubiquitous services: (1) the business logic analyst-designer focuses on the business logic of the system defining the application model without ubiquitous and technological consideration, (2) the context analyst-designer is responsible for the identification of ubiquitous functionalities supported by the system, (3) the adaptation designer defines a set of mechanisms and rules necessary for the dynamic adaptation of business logic to the context of the application at runtime, (4) the end user invokes services anywhere, anytime and anyhow. The system, transparently, provides the appropriate services using the context.

6. THE PROPOSED FRAMEWORK
A. Phase 1: Domain analysis
This phase is a preliminary study of the domain by analyzing both business and ubiquitous aspects to determine and identify business and ubiquitous requirements. The ubiquitous requirements are linked to business processes. The outcome of this phase is a domain model and a business use case model enriched with ubiquitous requirements which expresses the potential functionalities of the ubiquitous system.

![Figure 2. The proposed framework global view.](image-url)
Step 1: Identification of business requirements

The aim is to identify what the system performs and the necessary interactions with users, independently of their environment, technologies and equipment used. This step is performed by the business logic analyst-designer.

Step 2: Identification of ubiquitous requirements

This step is performed by the business logic analyst-designer in collaboration with the context analyst-designer. It consists in analyzing and studying the business requirements to extract and identify ubiquitous requirements.

B. Phase 2: Identification of business services

This phase is conducted by the business logic analyst-designer in collaboration with the context analyst-designer. The purpose is to identify a set of candidate business services from business processes identified in the previous phase. Our approach is based on the goal concept to identify business services. This detailed decomposition of goals enables to identify the goals of fine granularity (operational goals) as business services. This lead to obtaining a hierarchical goals decomposition.

C. Phase 3: Context modeling

It is realized by the context analyst-designer. It aims to formalize the context by analyzing the different ubiquitous goals identified in the previous phase and the definition of context model that all ubiquitous functionalities are based upon. The outcome of this phase is the context model. This phase is conducted in two steps: (1) defining ubiquitous rules where different ubiquitous goals should be detailed by transforming them into ubiquitous rules, (2) context model, where the set of ubiquitous rules defined allows the generation of context data to be considered in the system and to be modeled. The context model generated at this stage is a conceptual model that includes the entities, their profiles and relationships between them and with the environment.

D. Phase 4: Services description

In this phase we not only describe the functionality provided by the service, but also the context in which the service is executed. For each service, we must describe its functionality that meet user’s needs and the context in which the service is valid and can be executed. This phase is conducted in two steps: (1) description of service functionality and (2) description of service context.

E. Phase 5: Services adaptation

This phase is handled by the adaptation designer. It consists of defining a strategy for dynamic service adaptation to new contexts of use. As part of our work, we take advantage of the aspect-oriented programming [21] to extend and adapt the service behavior according to contextual changes, without modifying its business logic. Our adaptation strategy consists in considering the adaptation as an aspect. A ubiquitous service can have several possible behaviors according to contextual situations. These behaviors are encapsulated by a set of aspects which are attributed to it.

7. APPLICATION OF THE APPROACH ON A CRISIS MANAGEMENT CASE STUDY

7.1 Description of the case study

Crisis and disaster management is a special type of human complex organization [22] in which, heterogeneous actors belonging to different authorities need to collaborate and work together with the shared aim to solve, or at least reduce, the crisis situation. Each actor is equipped with different devices and communication technologies, and should carry on specific tasks. In addition, to the multiplicity and diversity of actors involved, the volume and the heterogeneity of information sources, the critical dependencies between actions and the dynamics of the situation make the management more complex. The underlying activities can be grouped into four phases: prevention, preparation, emergency management (response) and recovery.

In this paper we focus on the emergency management phase. In such scenarios, the process of response is divided into two phases: (1) Alert and (2) Intervention.

The Alert phase represents a significant level of emergency that informs required people depending on the scope and gravity of the situation. The intervention phase consists of taking a series of strategic tasks planned in advance to mobilizing quickly resources and elements required of various concerned organizations once an alert phase is identified.

7.2 Application of the approach

A. Phase 1: Domain analysis

The domain analysis phase collects general and global requirements. The identification of business requirements step allowed us to identify a set of business processes that have been modeled using BPMN - Business Process Modeling Notation (Figure 11).

![Figure 11. BPMN diagram corresponding to «Alert processing» Business process](image)

The identified ubiquitous requirements are linked with system functionalities as shown in Figure 12.
approach provides a high level of abstraction and has several advantages. Despite of the benefits of productivity and quality improvement of context and ubiquitous services development, the approach has the advantage of considering ubiquitous requirements to extract ubiquitous rules and identifying context information needed to build easily the context model using a predefined context meta-model. Ubiquitous requirements are developed in a separated way from system functionalities which is useful to enhance flexibility and facilitate system reuse and modifications. We applied this approach on a real case study related to the crisis management field.

9. REFERENCES

[17] Sheng, Q. Z. and Benatallah, B. ContextUML: A UML-based modeling language for model-driven development of context-
aware web services, In 4th International Conference on Mobile Business (ICMB’05), 2005, 206-212.


