Towards an approach for modeling and implementing pervasive services

Abdelhadi Bouain, Mohamed Nezar Abourraja, Abdelaziz El Fazziki, Mohammed Sadgal

Abstract

With the evolution of mobile technology (Smartphones, Tablets, 3G/4G connection, etc.), we are living a new era of ubiquity characterized by dynamic and complex environments composed of a multitude of artifacts that support mobility and perceive information from their environment. Several services can be offered in these environments, and users who do not have a technical vision of these services need only express their intentions to access them. In this paper, we propose an approach to describe and model pervasive services and the system architecture that implements these services starting from the concept of web services, given the advanced technology, standards, and the popularity of web services in the development of information systems.

Keywords: Context; Context-awareness; Intentions; Pervasive services.

1. Introduction

Rapid technological developments in mobile devices and 3G/4G internet connection have given new momentum to pervasive computing, which is intended to provide information to users any anytime/anywhere, using familiar objects[1], [2]. Currently, ubiquity is also seen as the ability to provide users spaces that offer a lot of services - ideally invisible way - and to access it, users will have to formulate their intentions (needs) - ideally using a natural language - without concern for technical details of the services, and will receive services that meet those needs and take into account their contexts[1], [3], [4].

In this paper, we aim to study the definition, design and implementation of pervasive services starting from the concept of web services, given the advanced technology, standards, and the popularity of web services. The rest of this paper is structured as follows. Section 2 tackles the issues of pervasive computing, pervasive service compared to web service, the context and intentions. Section 3 describes, in detail, our approach to model pervasive services; this approach is based on a MetaModel for describing the functional part of pervasive services, intentions, the context and the interactions among these elements. The system architecture that implements the proposed approach is detailed in Section 4. Section 5 is used to conclude this work by presenting a view of the untreated axes in-depth and our future works.

2. Related Work

2.1. Pervasive computing

Marc Wieser had the dream of living in a smart dust, surrounded by a set of invisible devices that offer information to users anywhere, anytime. These devices can form networks and get information without any action of a user -Pro-activity- [2].
The paradigm of pervasive computing has evolved with technological advances experienced by wireless sensors, 3G/4G network, and Smartphones that are increasingly endowed with sensors (accelerometers, gyroscopes, barometers, light sensors, proximity sensors, etc.). This technological evolution has led to a change in the trends of pervasive computing, which is becoming more focused on collecting context information and use of mobile devices [1], [4].

Pervasive computing can also be regarded as the ability to offer the users services tailored to their environments and their needs with the minimum interaction [1], [3], [5]. The paradigm of pervasive computing based services was introduced for the first time by Zhou et al. [6], with the aim of using web services to create a pervasive environment given the modularity and weak coupling between web services provider and user.

2.2. Web services Vs. Pervasive services

Web services are one of the popular technologies used to distribute information between the provider and the user, this technology standardized by the W3C( World Wide Web Consortium). A web service provides features to clients using an interface based on input and output parameters [7]. But in a pervasive environment a number of principles must be adhered, such as invisibility and efficient use of space which can be approximated by the minimal interactions between users and services [2]; context-awareness, that is, the acquisition and use of any information in the execution environment that may influence the processing performed by a service [1], [8]–[10]. Therefore, pervasive service and web service are two different concepts. In few words, a pervasive service is an entity that offers features tailored to the user's context and intentions [3].

2.3. Context

2.3.1. Definition

Several definitions have been given in the literature for context [8], [9], [11]–[14]. In our work, we consider context as any information in the execution environment that may influence the processing performed by a service.

Context-aware applications (Services aware) are applications (services) that dynamically adapt their operations and their interactions with users according to the changes in their execution environment [11], [14]–[16].

Context adaptation might be several levels: Context-based adaptation, that refers to services or applications that request information from their environment synchronously and adapt their operations based on this information. Context-aware adaptation, which refers to applications that observe their environments and impose their conditions on certain types of information; once these conditions are met, a notification is sent to the application that will trigger the corresponding action. Situation-aware adaptation is the most complex context adaptation. Applications require a set of information about the current situation and depending on the availability of this information, several execution scenarios are possible [11].

2.3.2. Context modeling approaches

Proposed approaches for context modeling may be categorized as follow:

**Key-Value Models**: This approach presents the context in the form of pairs (key, value). The advantage of this approach is its simplicity, but it lacks in structure and is inefficient to describe complex context information [11], [17].

**Markup Schema Models**: This approach represents the context in a hierarchical format using language derived from SGML, like XML. We can take, as an example of this approach, CC/PP (Composite Capabilities / Preferences Profiles) [18]. Using Markup Schema Models provides an opportunity to describe and structure the context and also provides partial validation of context.
information that must fit the description of the language used, but this approach does not solve the problem of the ambiguity and incompleteness of information [11], [16], [17]

**Graphical Model:** This approach uses graphic symbols to model context information. We cite, for example, the Henricksen proposal [19], which is an extension of Object-Role Modeling and Sheng proposal (ContextUML) [20] which is a UML extension. The Graphical Model approach ensures a good representation of the context, but it is less formal than other approaches [11], [16].

**Object Oriented Model:** The objective of this approach is to take advantage of the mechanisms offered by object-oriented (Inheritance, encapsulation, polymorphism, etc.) to model the context [11], [21]. Approaches based on the object-oriented model provide a context abstraction and facilitate the sharing and reusing of the context, but only between specific types of applications.

**Ontology-Based Models:** Among the approaches that have used ontologies to model the context, we refer to CoBrA (Context Broker Architecture) [11], [21], [22].

**Logic-Based Models:** a formal approach, based on the logic with the aim of defining rules or conditions that may infer facts or expressions from other facts or expressions. Approaches based on the logic model provide a mathematical formalism and make easier the reasoning about context information, but they consume a lot of system resources and time to make inferences about the various predicates [11], [16], [21].

2.4. Intentions

2.5.1. Definition

Users in pervasive environments are not necessarily computer experts; they express their needs as objectives to be achieved without concern for the description or implementation of services; for example, convert a sum of money in foreign currency, consult unpaid invoices, make reservations in hotels, etc. These objectives expressed in natural language are conventionally named "Intentions".

2.5.2. Modeling intentions

To reduce the gap between the technical level of services and user level, users must be given the opportunity to formulate their queries by expressing only their needs or goals to achieve, using natural and easily understandable language. Among the proposed approaches to achieving this goal, we quote the proposal of [23]–[25] which decomposes goal into a verb, target and parameters. For instance, book a room in Marrakech next weekend may be seen as follows: (Book) is the verb; a room is the Target; in (Marrakech next weekend) is a parameter. The target may be either a product or result; parameters can represent the place, the manner, the quality, the quantity, the time, the beneficiary, etc. Therefore, Reserve a room in Marrakech can be noted as follows: (Book) verb; (a room) product (in Marrakech) place (next weekend) time.

In general, describing an intention is tantamount to describing the action that led to this intention. That is to say, for modeling an intention, we must identify its constituent elements by responding to questions such as: what, how, and why [23]–[25].

3. Proposed approach

The integration of context and intentions in services will provide pervasive services that meet users' needs - A need expressed in non-technical language and, ideally, in the natural language spoken by users - taking into account all environmental factors (context) of the user.

To describe pervasive services, we propose a generic MetaModel for describing the service elements, context elements, intention elements, and relationships existing between them (see Figure 1) this MetaModel is proposed on the basis of several works at the level of context
modeling and intentional services [16], [17], [20], [23]–[26]. The elements of this proposed MetaModel can be divided into four categories (See Figure 1).

The service elements: As for web services, pervasive service offers a set of operations that communicate using messages. A service can be an atomic or aggregate of several atomic services. Aggregate services are of two types: composite services and variation services. Composite services use the logical AND operator between atomic services to make the aggregate service. Variation services provide the choice to achieve the aggregate service using the logical OR operator (See Figure 1).

The context elements: The context elements describe the structure of information extracted from the execution environment.

Context Information: represents the context information that is defined by its value, its unique identifier (to facilitate research), a Timestamp (to manage the history), a validity which is useful in environments experiencing a perpetual change in context information, and a description that is a set of terms to describe the context information (See Figure 1).

Simple Context: Context information can be simple or composed. Simple Context represents simple and clear information that does not need other context information to complete it.

Composed Context: It is context information which is composed of other context information; for example, a complete medical report is a set of context information extracted from various sensors (glucometer, blood pressure, electrocardiogram “ECG” etc.). Another example is the user's mental

Figure 1. Generic MetaModel for Pervasive Services
state that is not simple context information to deduce because it requires the combination of several other context information extracted from various sources e.g., camera (emotions), microphone (tone of voice), messages posted by the user in social networks, etc.

Source: an abstraction of sensors or other sources that provide context information. The sources can be either simple or composed of several other sources of context; for example, the magnetometer, accelerometer, GPS, etc. are simple context sources that can be part of a Smartphone that can be regarded as a composed source.

Category: the context information can be classified into categories for a structured view to facilitating research and identification of this information.

Context Binding elements: The Context Binding elements make it possible to implement the mechanism of context awareness; that is why we made the distinction between applications that are context-based and those that are sensitive to context (context-aware). Context-based applications use the context information as input parameters, while context-aware applications observe the environment and adapt their actions according to its state. The binding mechanism between services and context information (Context Binding) offers two options: Simple Binding to directly use the context information in services and trigger binding which allows proposing services to users based on conditions applicable to the context information.

Intentions elements: A service aims to achieve an intention starting from an initial situation to attain a final situation. The Pre-conditions are conditions that must be checked before running a service. The post-conditions are conditions that must be checked at the end of service execution. Each intention has three basic elements: the goal to be achieved (What), the means to use (How) and the reason of this intention (Why). For example, to distract and relax, a customer books a flight ticket to Marrakech using his credit card. The goal is to book a flight ticket to Marrakech; the means is the credit card; the reason is to distract and relax. The goal in its turn is composed of a verb; target and parameters [23]–[25].

4. System Architecture

To implement an information system based on the pervasive services we present in this section a view of the architecture of this system. This section consists of three subsections; the first is dedicated to the implementation of context; the second is for the implementation of intention and the third subsection presents an overview of the system.

4.1. Context implementation

In order to achieve context-aware systems, we must take into account several aspects of context such as collection, storage, sharing context information, etc. Several platforms for context management have been proposed; some are implemented using web services as ESCAPE [27] COWSAMI [28], inContext Project [29], etc. Others use Java technology, especially RMI (Remote method Invocation) such as JCAF (Java context awareness framework) [30] or CORBA (The Common Object Request Broker Architecture) as GAIA Project [31], but none of these platforms cover all aspects of the context [11], [14], [32].

To integrate the context at the pervasive services, we adopt a multi-layered approach [11] . The first layer is dedicated to collecting information from the environment; the second layer is dedicated to adaptation and reasoning about context information; the third layer is interested in storing context information; the fourth layer's role is to ensure context information discovery, while the last layer is the application layer which represents the use of context in pervasive services.(See Figure 2)
The advantage of using a multi-layered approach lies in the fact that each layer uses the information from the layer below without having to consider the detail of how this information was obtained, which encapsulates the complexity of different layers.

4.2. Intentions implementation

The objective of the intentions is to present to users appropriate services that meet their needs, a need expressed using non-technical terms but business terms. The description of the pervasive service supports the intentions (See Figure 1) what is lacking is the mechanism which uses context information and users’ intentions to create queries able to identify the good services that satisfy the specific needs of users. We propose to use a process of formulating queries. Inputs of this process are user intentions expressed in natural language (expressed in terms of objectives to be attained, context information stored at the storage layer and domain-specific ontology to generate queries able to identify services that best meet user needs [3] (See Figure 3).

Among the approaches that use ontologies to reformulate queries expressed in terms of goals, we cite [23]–[25]. This approach offers two types of ontology: Ontology of verbs and Ontology of products. The Ontology of verbs represents the syntactic and semantic concepts related to verbs used in the query (intention) of the user. Ontology of products is a domain-specific ontology that describes all the inputs and outputs of services (the objects manipulated in the domain). Users’ feedback can also be used to improve the process of reformulating queries.
4.3. **Overall architecture of the system**

The overall system architecture is shown in Figure 4. In fact, pervasive services publish their descriptions at a directory by specifying the full functionality and other information necessary to invoke them. Users formulate their intentions that will be sent to the process of reformulating queries to generate queries able to locate good services that meet user needs and take into account their contexts. Once the request is made, it is addressed to the directory that returns the corresponding service description to allow the user to invoke it.

Once Users have consumed a service or set of services, they are invited to give feedbacks that will be used to improve both the process of query reformulation.

**Conclusion**

In this article, we presented the pervasive services compared to the web services, which helped us to identify context and users' intentions as two elements that can reduce the gap between these two concepts. The analysis of the notion of context and intention led us to propose a metamodel for pervasive services. After discussing its elements in detail, we presented an architecture for implementing pervasive services based on this proposed metamodel.

The proposed approach makes it possible to have environments with a multitude of services, and to access them, users have to express their needs in a natural language without worrying about the technical implementation of these services or query language to use. The result returned to users is a set of services that meet these needs, taking into account users' contexts. Context processing and reasoning are managed in a centralized way, which can be seen as a disadvantage, but the implementation of the processing and reasoning process at the level of clients' devices is very expensive, and can be very difficult in open environments experiencing various, often occasional devices.

Our future work will concentrate on other areas that were not covered in depth in this article, such as security, services composition, the publication and discovery of pervasive services. Our ultimate end...
is to develop pervasive information systems which provide users with services that meet their needs and are appropriate to their context. [13]

References

Towards an approach for modeling and implementing pervasive services
Abdelhadi Bouain, Mohamed Nezar Abourraja, Abdelaziz El Fazziki, Mohammed Sadgal


