Introducing contextual objects in an adaptive framework for wide-area mobile computing

A.-M. Kermarrec[‡], P. Couderc[†], M. Banâtre[†]

† Irisa/Inria
‡ Irisa/Université de Rennes 1
Campus Universitaire de Beaulieu
35042 Rennes Cedex, France

e-mail : {akermarr,pcouderc,banatre}@irisa.fr

Abstract

The emergence of mobile computing combined with the increasing popularity of the Internet has introduced new challenges that are not addressed in traditional distributed systems. In particular, mobile computing is characterized by the extreme variability of the environment components such as network bandwidth, computing power, location, services available, etc. The goal of our work is to provide a framework to dynamically adapt applications to the continuous changes of mobile environments including physical constraints and more qualitative changes. As a basis of our work, we present in this paper the concept of contextual objects. This approach allows to express the dependencies of an object according to a number of contexts such as the time, the location or the physical aspects (available network bandwidth or computing power) in a changing environment.

1 Introduction

In recent years, mobile computing has emerged as a promising class of systems and its importance is still growing: the increasing availability of wireless technology and mobile devices allows users to move around with their own computing environment. At the same time, the number of Internet users grows exponentially and facilities for accessing Internet are henceforth widespread available thereby increasing the need of ubiquitous network access. Supporting mobile users on the Internet is thus becoming essential and leads to new challenges which are not addressed in traditional distributed systems [8]. Although in a static environment system configuration and available resources remain unchanged, mobile computing is mainly characterized by the extreme variability of any kind of computing resources. To deal with the dynamic nature of mobility, tools allowing to adapt applications to such variations at any level (network, systems, resources and software) are needed.

Many works have been proposed to achieve transparently either the masking of mobility at the network level or at the system level, or the adaptation to physical constraints. Most of these proposals are able to adapt themselves to mobile constraints but are not specific to mobility. In particular, they do not exploit environment informations. Context-aware applications have been introduced for ubiquitous computing [12] in order to exploit further the mobility. Nevertheless, such applications are usually restrictive to the location criterion and only applicable in local-area environments.

As users move across the Internet, their environment changes. Some new services become available, some objects remain unchanged whereas others may be no longer relevant. Mobile computing thus leads to a new class of *context-dependent* applications where objects may have different value depending on location, support of connection, current network bandwidth, etc. Thus, adaptation is the key to mobile computing. Our goal is to provide a framework to develop *context-dependent* applications for mobile users. As users move, the support should reflect the real-world environment in terms of relevance of the downloaded information, performance of provided services, and resource constraints. The contribution of our approach is that instead of simply masking the mobility and addressing the adaptation of environment changes as a degradation, we aim at taking benefit of the mobility.

To address this issue the first important step is to propose a way to express clearly the dependencies of an object according to several criteria which may vary as users move. In this paper, we introduce the concept of *contextual objects* as a basis of our adaptive framework.

We first briefly present existing approaches to mobility. We then introduce contextual objects as a way to take into account mobility of users across the Internet. We give examples of both contextual data and code. We then address open problems introduced by considered applications. Section 5 concludes and give future research directions.

2 Existing approaches to mobile computing

A mobile environment represents a computing environment involving mobile entities. A mobile entity may be a portable computer, a personal digital assistant (PDA) or a user moving and connecting from place to place. Compared to traditional static systems, the most obvious difference in mobile computing is that resources needed to run an application and system configuration continuously vary from time to time and place to place. Effectively, a mobile device may experience important variations of physical resources such as network bandwidth or computing power as it moves from one site to another. Many works have been proposed to accommodate these resources constraints in mobile computing. We distinguish between two approaches: masking-based and adaptation-based approaches.

Masking-based approaches aim at providing a mobile device with the same set of services as on a desktop system. They may be implemented at the network or operating system level. Network masking approaches consist in emulating the behavior of the network protocols of a static environment. For example, Mobile-IP solutions [7] allow mobile devices to keep their own IP addresses and do not require any modification to applications. Masking-based approaches implemented at the operating-system level allow applications to run unmodified in a mobile environment. For example, the CODA file system [4, 5] aim at masking disconnection periods by caching and hoarding techniques. However, these approaches are not completely satisfactory since it is not always possible to mask everything to applications. For example, Coda requires some assistance at the user to solve update conflicts. Moreover, many adaptations can be handled only by the applications themselves. For instance, degrading gracefully image quality with respect to user's requirements and according to a network bandwidth is clearly not possible at the system level.

Adaptation-based approaches partially solve these latter drawbacks. Some environment informations are provided by the application in order to enable them to adapt their behavior and their quality to the changing conditions. For example Odyssey [6] requires minor operating system changes to allow the adaptation of applications to resources changes.

However these approaches are restrictive since they are limited to degradation of applications behavior. In particular, they only deal with low-level constraints monitoring and do not help the applications to choose which kind of adaptations are the best. As these approaches do not take benefit from the mobility, ubiquitous computing [12, 10] addresses this issue. Context-aware systems [11, 6, 13], such as Parctab [9] or Dataman [1], have been developed in this context and exploit the local environment to adapt existing services depending on the local resources and to provide new services. However, in these approaches, contextual information is closely related to the physical environment such as nearby people or resources (a printer for example). Although ubiquitous computing exploits physical local context to allow a better interaction with local objects in a limited area such as a floor, a building or at most a campus, we consider mobile computing across the Internet. Such a dimension introduces challenges that a central management scheme can no longer solve. In addition, we also want to address adaptation to physical constraints such as adaptation-based approaches do.

3 Contextual objects

To take into account context-dependent objects we introduce *contextual objects* where objects, named independently from the context, may have different values depending on the context of the application using them. Our approach may be considered as an aggregation of ubiquitous computing extended to wide-area systems and adaptation-based approaches to deal with physical constraints. The dependencies of an object are expressed according to both physical constraints and more qualitative criteria such as location or relevance of the retrieved information.

In traditional distributed systems, an object, which may take different forms depending on the considered application, is defined as an identifier used to manipulate information or to execute a program independently of its current value which may vary with time. The implicit variation domain is time, since it is the only parameter used by the system to differentiate object's state transitions. Likewise, most features related to data management, such as coherence, location, or fault-tolerance, are based on this temporal nature of objects changes. Under this assumption, at a given time a variable shared by several users reflects the same value (actually, approximatively in distributed systems). For a generic name, an object may be replicated at different places and may have different values if the coherence is not strictly maintained. However eventually, replicas should converge to the same value.

Characterizing objects changes only with respect to time is no longer sufficient to take into account the dynamic nature of system's state introduced by mobile computing. In this paper, we introduce the concept of *contextual objects* as a basis of our adaptive framework. This new way of characterizing objects in wide-area applications allows to deal with the highly dynamic aspect of mobile computing and to express the context-dependent characteristics of objects.

We distinguish between *quantitative* and *qualitative* changes in a mobile system. Quantitative factors refer to as physical constraints such as the computing power, the available memory or the network bandwidth. They can be easily monitored by a system. A number of existing approaches deal with these resources constraints such as Rover [3] or Odyssey [6]. On the other hand, we define qualitative factors as those that make sense from an application point of view. For example, location may be a qualitative factor for a geographical object: the value of a location-dependent object does no longer make sense when a user accessing it moves. Nevertheless, only the application is able to detect the impact of such qualitative changes.

A contextual object represents the interface to the contextual informations of the environment. A contextual object mainly consists of:

- **An object identifier**: the notion of identifier for a contextual object is more complex to define than in a traditional object system since a contextual objet may refer to different real objects, possibly named in another framework. However, we assign an unique identifier to a contextual object. By unique, we mean that in a given context, two contextual objects cannot have the same name.

- **A context-sensitivity list**: this list describes the dependencies of an object and the kind of context to which it is sensitive. As users move, part of the environment changes and the current object may be reevaluated according to these changing conditions. A context-sensitivity list may also be made up of others contextual objects. This feature allows to express relations between contextual objects.

- **A real object reference**: this part of the conceptual object is a reference to some real object which represents the value of the contextual object in the current context. Separating the contextual management from the real objects allows an easy integration in existing information systems like the Internet (HTML page, FTP loadable file, Java class or a GIF picture). We refer to such an already existing resource as a *legacy* object.

The adaptive framework has to select the right value according to a given context as quantitative and qualitative variations occur.. Figure 1 depicts an example of three contextual objects each dependent from a different criterion. Figure 1.(a) represents a time-sensitive object, the representation is here exactly the same as in a traditional object model. For example an object may be replicated at two different places with a different value (value at time T_1 and value at time T_2) if the coherence has not been strictly maintained. The goal of this example is to show that our model integrates time-dependent objects which is the usual way of representing objects. Figure 1.(b) represents a bandwidth-sensitive object. Consider for example an Image, depending on the available bandwidth it would be much more relevant or more efficient to retrieve a black and white image or a colored image. Finally, Figure 1.(c) represents a location-dependent object. A weather forecast information is an obvious example. Depending on the changes of the environment and the context sensitivity of an object, the framework retrieves different legacy objects for each object identifier.

One of the major contributions of such a model is that it may be applied to both code and data in order to adapt the service delivery to mobility with respect to relevance and performance: relevance since an information must be coherent with a given context and performance since efficiency may be improved by taking into account local context (communication support or close replicas).

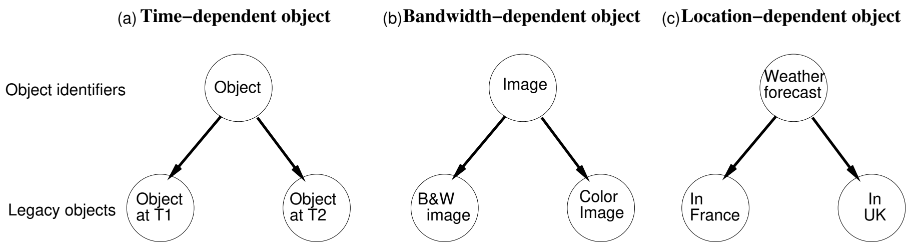

Figure 1: Contextual objects

3.1 Contextual data

To illustrate, consider the example of a newspaper-based distributed information system that provides access to various multimedia services including an electronic newspaper [2].

The concept of contextual data may be particularly useful in such a context. We focus here on location-dependent data where the context mirrors the geographical location. *Location-dependent data* have different values in the spatial domain at a *given time*. Note that the spatial dimension is added to the temporal dimension and does not replace it. Hence the variation domain is two-dimensional.

Consider a French user subscribed to two services of an electronic newspaper: news and theater schedules. If he moves from Paris to Amsterdam, he would like probably still access to French news whereas getting Parisian theater schedules is no longer relevant. The news service is then considered only as a time-dependent object whereas the theater schedule service is referred to as a time and location-dependent object since its value may change dynamically depending on the mobility of users (Figure 2). When the mobile user connects in Amsterdam, our framework has to detect the change of context and to react by delivering French news and Dutch theater schedules. We could also add a linguistic criterion, and increase the dimension of the context-sensitivity list, to the theater schedule context such as the French user gets the Dutch theater schedule in French rather than in Dutch when it is possible.

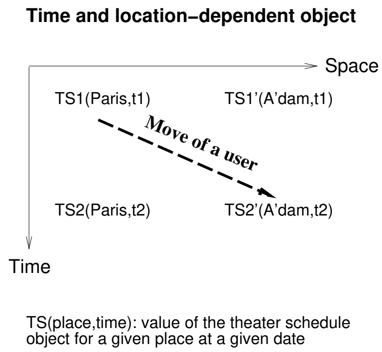

Figure 2: Two-dimensional variation domain

3.2 Contextual code

To illustrate the contextual code paradigm, let us stick to our multimedia example. We consider that a mobile user may receive either the complete video sequence for a given service, or text and colored pictures or possibly degraded pictures and text depending whether the hardware and communication bandwidth is high or not. The framework must be able to adapt the computation of the electronic newspaper for a user connected from a silicon graphics to a mobile phone. The concept of conceptual code may also be particularly efficient to adapt the code of an application to different terminals depending from where a user is connected.

The context information is also particularly useful to improve performance while executing a program. In many cases the context is derived from the location of a user or an application and a local service may be used for more performance. To illustrate, consider the execution of a Java-based application. Classes need to be dynamically loaded from the source. This can be very expensive when a user has moved far away from the originating site of the class and is henceforth connected remotely from his starting point. When the change of context is detected, Java classes may be dynamically loaded locally thus improving performance. Whereas specifications are similar, implementations may be different [1]. The system may need to adapt the whole application to this change of one class implementation. The relation between classes used by an application can be intregrated in their context-sensitivity list. This adaptation is essential in a wide-area environment since update of code is not necessarily maintained i=on the Internet because of its large dimension.

We are currently investigating the implementation of an experimental framework for Java applications using Java *class loaders*. Class loaders, that may be user-defined, enable Java virtual machine to load classes dynamically. Such a user class loader can load the class implementation data from an alternate source. Java class loaders can use contextual information to load dynamically the most appropriate class in term of relevance/performance.

4 Open problems

There is a number of features that a framework supporting contextual objects may provide both for user interface and for application development.

- **Definition of a context**: First of all, our framework must provide applications with a way to express clearly the context of an object that is to say the number of criteria to which it is sensitive.

- **Dynamic detection of context**: Our framework must also be able to detect dynamically and efficiently changes of context. In order to anticipate reactions to mobility, users information such as agenda or past behavior may be useful. To allow an easy and efficient adaptation, mobility must be integrated in the system as a standard operation. We are currently investigating the detection of a change of location for a mobile device using mobile-IP. This step must also exploit the mobility to provide the application with possibly new services.

- **Adaptation**: For a given context, our framework must be able to apply a selection algorithm to retrieve the right value of a contextual object (data or code) depending on users requirements and local context. An application may have a different behavior depending of the retrieved value of its contextual components. For example an electronic newspaper could range from a multimedia version to a simple text to be able

[1] Actually, such situations are not currently allowed with Java but a JVM could be easily modified to allow different implementations for a common interface. This feature is useful for an adaptive framework

to adapt to different mobile devices. The major problem is to ensure the coherence of the global application when the value of one or several contextual objects change. In contrast to the temporal domain where a invalidated value is no longer useful, a location-dependent value invalidated by a move may be accessed again if a move back occurs. Then context-limit must be attached to standard coherence information. Contextual objects thus lead to a new way of investigating coherence issue.

  Contextual objects could be handled by *contextual servers*. Each contextual server manages a set of contextual objects. The main task of the server is to analyze the dependencies information from the context-sensitive list and to notify the framework or the application of relevant changes.

- **Exploiting mobility**: The last issue we are currently investigating is how to exploit mobility to improve wide-area issue by using contextual objects. As devices or users move, the value of contextual object change. Mobile users may then be used to carry information from place to place, and act as a high-bandwidth, low-latency communication medium. As users move (and if we consider a mobile system, users move anyway), the information they carry may be used to implement weak coherence (data and code) or fault-tolerance.

5 Conclusion

We introduced in this paper a new way of characterizing data depending on their context. Our approach aims at providing an adaptive framework to ensure transparently the delivery of a service, or the right execution of an application depending on a local context as users move. Our new way of representing objects (code and data) helps in taking into account mobile computing for highly adaptive applications. This representation integrates:

- the traditional way of representing objects according to time;
- a way to express reaction to quantitative changes (different values of an object depending on physical constraints such as in adaptive-based approaches);
- a way to express reactions to qualitative changes (different values of an object depending on application criteria such as in ubiquitous computing).

Our approach integrates adaptation-based approaches and ubiquitous computing by allowing applications to use in a transparent way, objects appropriate to a given context. Moreover, our approach gives a way to take into account from the start the contextual aspect of objects involved in mobile computing. Another original aspect is the use of contextual code either to improve performance or to improve adaptation to hardware constraints.We are currently investigating the impact of such an abstraction on detection, anticipation, coherence or fault-tolerance issues. Our electronic newspaper application, used in a mobile environment by users possibly connected from different devices, reflects the need to adapt to changes of time, physical variations and location changes.

References

[1] A. Acharya, T. Imielinsky, and B. R. Badrinath. Dataman project: Towards a mosaic-like location-dependant information service for mobile clients. Technical Report TR-320, Rutgers University, New Brunswick, 1994.

[2] M. Banâtre, V. Issarny, and F. Leleu. ETEL: A Newspaper-Based Distributed Informations System. In *Procs of the 7th ACM SIGOPS European Workshop: Systems Support for Worldwide Applications*, Connemara, Ireland, September 1996.

[3] A. D. Joseph, A. F. deLespinasse, J. A. Tauber, D. K. Gifford, and M. F. Kaashoek. Rover: A toolkit for mobile information access. In *Proceedings of the fifteenth Symposium on Operating Systems Principles*, December 1995.

[4] J.J. Kistler and M. Satyanarayanan. Disconnected operation in the coda file system. *ACM trans. on Computer Systems*, Vol.10(No.1):pp 3–25, Feb 1992.

[5] P. Kumar and M. Satyanarayanan. Flexible and safe resolution of file conflicts. In *Proc. of the USENIX Winter 1995*, January 1995.

[6] B. D. Noble, M. Satyanarayanan, J. E. Tilton, J. Flinn, and K. R. Walker. Agile application-aware adaptation for mobility. In *Proceedings of the 16th Symposium on Operating Systems Principles*, October 1997.

[7] C. Perkins. Ip mobility support. RFC 2002, October 1996.

[8] M. Satyanarayanan. Fundamental challenges in mobile computing. In *Proceedings of the Fifteenth Annual ACM Symposium on Princiles of Distributed Computing*, May 23-26 1996.

[9] B. N. Schilit, N. Adams, R. Gold, M. M. Tso, and R. Want. The parctab mobile computing system. In *Proc. of the 4th Workshop on Workstation Operating Systems*, pages pp 34–39, Napa, CA, October 1993.

[10] B. N. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *IEEE Network*, 5(8):pp 22–32, September/October 1994.

[11] G. M. Voelker and B. N. Bershad. Mobisaic: An information system for mobile wireless computing environment. In *Workshop on Mobile Computing Systems and Applications*, 1994.

[12] M. Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36(7):pp 74–83, July 1993.

[13] G. Welling and B. R. Bahdrinath. A framework for environment aware applications. In *Proc. of the International Conference on Distributed Computing Systems*, Baltimore, Maryland, may 1997.