ISAM: a Software Architecture for Pervasive Computing

Adenauer Corrêa Yamin
Computer Science Department
Catholic University of Pelotas
Pelotas, RS, Brazil
adenauer@ucpel.tche.br

Jorge Luis Victória Barbosa
PIPCA - Unisinos
University of Vale do Rio dos Sinos
São Lepoldo, RS, Brazil
jbarbosa@unisinos.br

Iara Augustin
Computing and Electronics Department
Federal University of Santa Maria
Santa Maria, RS, Brazil
august@inf.ufsm.br

Cláudio Fernando Resin Geyer
Informatics Institute
Federal University of Rio Grande do Sul
Porto Alegre, RS, Brazil
geyer@inf.ufrgs.br

Abstract

The Mobile Computing is a vision of future that promises to modify the way people work and interact today. It has emerged as a new field, distinct from conventional distributed computing by its focus on mobility and its consequence. The physical and logical mobility allows the creation of applications with news requirements: built-in mobility, adaptability and flexibility. So, software production for this global mobility is complex. Challenges include designing applications which service levels and available resources are unpredictable, and the behavior is variable in time and space. To provide a system architecture that simplifies the task of implementing the mobile applications with adaptive behavior is the objective of the ongoing ISAM Project. In order to achieve its goal, ISAM uses as strategies: (a) integrated environment that provides a programming paradigm and its execution environment; (b) uniform treatment of adaptation through multilevel collaborative model, in which both the system and the application contribute for that. The scheduling process – the core of ISAM architecture, manages this collaboration. Introducing HoloLanguage abstractions that express adaptability makes the control of adaptation by part of application. So, adaptation is not fully transparent to applications. In this paper, we present the main decisions used for designing the ISAM Architecture. We also introduce a first taxonomy for mobile applications with adaptive behavior.

Keywords: mobile computing, middleware, distributed mobile applications, adaptive behavior, taxonomy for mobile applications, platform for pervasive computing.

1 Introduction

Mobile computing is a new computational paradigm emerging from wireless networks and distributed systems technologies. The mobility allows the existence of various scenarios, from nomadic computing to pervasive computing [3]. In the last scenario, the users carrying portable devices, such as palmtops and notebooks, have access to shared infrastructure independently from their physical location or displacement form. The growing availability of communication facilities permits the shift of the mobile applications from a personal administration perspective to a more professional usage. Considering the viewpoint of pervasive computing, the computational power will be available everywhere [17].

Environments inherently mobile have features of heterogeneous processing with frequent and fast fluctuation in service quality provided by networks and in variability of resources access and availability. The systems operate connected to several networks with variable connectivity levels, like high speed as well as low speed and also network disconnected. As a consequence, mobile software production is complex and its components change in time and space in terms of connectivity, portability and mobility. To alleviate the impact of these changes, the application must have an adaptive behavior.

The adaptation is a fundamental concept for mobile computing [25]. However, adaptation is not easy for implementing. Traditional distributed systems were created based on suppositions about environment, such as permanent connection and resources availability, which is not true in the mobile environment. This fact prevents the direct usage of solutions from distributed system area. We believe that new mobile systems architectures are necessary, modeled with built-in mobility, flexibility and adaptability. With this motivation the ISAM Project[1] (*Infra-estrutura de Suporte às Aplicações Móveis* – Support Infrastructure for Mobile Applications) is going on.

The project involves four universities. It aims the provision of infrastructure for developing and executing distributed mobile applications. Our system is designed to support multilevel collaborative adaptation at application level and system level. In ISAM, the system adapts to provide quality, while the application, if necessary, adapts until the lowest quality level tolerated by the mobile user. The innovation of this middleware is the uniform design of adaptation and its suitability for general-purpose mobile application.

This article describes the ISAM architecture, and the remainder paper is organized as follows. Section 2 introduces the scope where ISAM is inserted. The ISAM middleware architecture, its abstractions and main entities, are described in Section 3. The main decisions for architecture design are discussed in Section 4. Section 5 proposes a first taxonomy for mobile applications with adaptive behavior. Following, Section 6 discusses some actions of the scheduler and of the application providing a multilevel collaborative adaptation. Section 7 reviews the related work. Finally, we give some concluding remarks in Section 8.

2 Scope ISAM

The term “mobile computing” is still not well defined. Various computational environments use this denomination for indicating some kind of mobility [5][7][9][10][11][16][17][18][19]. For us, mobile computing is “the distributed computing which elements location changes during the computation execution.” Some scenarios derive from this definition depending on the element with mobile capacity: hardware – wireless computing; user – nomadic computing; program – mobile code; computation - mobile agent. The most general scenario that join all them is the provided by pervasive computing, which allows physical and logical mobility while it maintains the global network connection. This last scenario is the focus of ISAM Architecture.

[1] ISAM Project is supported in part by FAPERGS Foundation. The others partners of the consortium are: Federal University of Rio Grande do Sul, Federal University of Santa Maria, Catholic University of Pelotas and University of Vale do Rio dos Sinos – Brazil.

2.1 Why to adapt?

The characteristics of mobile computing are portability, mobility and connectivity [3]. These characteristics insert constraints in the environment. A portable computer is small, light and it requests sources of energy of little power. As a consequence, portable computers have restrictions in memory size, storage capacity, computational power and user's interface. Furthermore, the portability increases the loss risk, or damage of the mobile device. When in movement, the mobile device changes its location and possibly its network contact point. This dynamic nature inserts questions about node address, user location and location-aware information. The wireless connection raises other obstacles: intermittent communication (frequent disconnection, block signaling, and handoff), restricted and highly variable bandwidth, high latency and high error rate.

Another aspect to consider is the kind of network connection. Research, initiated in Coda [25], has shown that three operation modes are present in a mobile system: strongly connected – when using a cabled network; weakly connected – when using a wireless network; and disconnected. The disconnected mode is voluntary and used to save mobile resources and power consumption. So, the mobile device operates in this mode most of the time.

Besides, other three aspects of the distribution become very important when mobility is present: scalability – inserted by potential size of users set; heterogeneity – inserted by different environment, devices and networks; dynamism – inserted by change in the resource availability and by user movement. All these aspects are not isolated, and the system architecture must include them.

These restrictions are inherent to the mobility nature, and they introduce new demands in the design of mobile applications [11][14][15]. The mobile applications must be more flexible than current distributed systems when one resource is unavailable/inaccessible or when the availability/accessibility level of resource was reduced. To provide the best possible performance within the user's expectations, these systems must explore the adaptation concept. The solutions of the traditional middleware for distributed system are not appropriate for that, because dynamic adaptations are not allowed. Another problem has its source in the fact that distributed systems were designed with suppositions, such as permanent connection and resources availability, which are not true in the mobile environment. Besides, the majority current solutions satisfy the needs of specific application, and are not appropriate to reuse in general-purpose applications [9].

We believe that there is a fundamental need for new mobile software architectures that are adaptive in nature. So, the *adaptability* is the fundamental idea of the ISAM architecture. Both the applications and the system were projected with adaptation in the mind.

2.2 Pervasive Environment ISAM: ISAM_PE

To develop distributed mobile applications, we consider that mobile hosts must use the existing and accessible fixed network infrastructure, and they have to take advantage of the environment such as Internet. This environment model is reflected in the basic elements of execution system ISAM, present in Figure 1:

- EXEHDAcell – It's the acting area of EXEHDAbase. It is composed by EXEHDAnodes;
- EXEHDAbase – It's the initial contact point of mobile host with the ISAM services resident in fixed network. It owns the identification, authentication and activation functions of the system;
- EXEHDAnode – they are the system nodes, responsible for distributing mobile application execution and processing services of the ISAM management;

- EXEHDAmob-node – are the system mobile nodes responsible for the user interface services and small management functions. They are similar to EXEHDAnode and they act according to their computational power;

- EXEHDAhome – It's EXEHDAnode that works as a reference point for each mobile user in all networks.

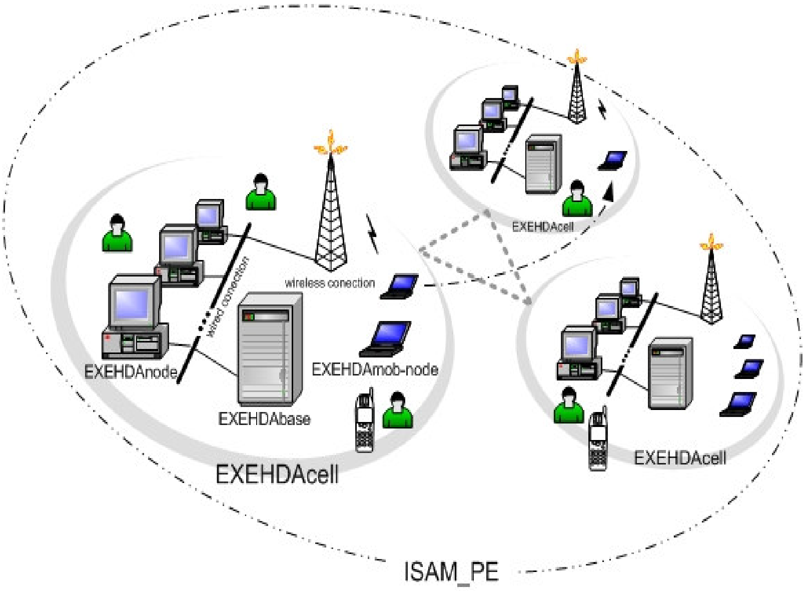

Figure 1: ISAM Environment Components

3 The ISAM Architecture

The challenge of adaptation is its complexity. At this point of the project, we are immersed in the design of collaborative behavior between system and application. The architecture is organized in layers with three abstraction levels, and it is directed to getting the maintenance of quality of provided service for mobile user through adaptation concept. Our approach for the adaptation problem in the mobile computing is illustrated in the Figure 2. For organizational matters, the ISAM architecture is presented through modules. The adaptation module is *virtual*, because it is presented in the conception of many others modules. In our work, we are interested in providing a uniform adaptive behavior in all system, whereas that adaptation is the critical point. Another important aspect in the architecture is the scheduling module – the core of ISAM architecture. Adaptation decisions, from application to scheduler, are based on the behavior profile of three entities: mobile user, application and the self-system, which is part of the execution context.

The higher layer is composed by distributed mobile applications, programmed with *HoloParadigm* abstractions [6] that express mobility in a natural way. *HoloParadigm* is being modified to adding new abstractions that allow expressing adaptability at application level (ISAM Adapt), whereas the lower layer is composed by existent distributed systems, such as the native operating systems and the Virtual Machine Java. Furthermore, the ISAM architecture was conceived assuming the existence of a mobile network infrastructure (wireless network + cabled network), which provides support to accessing network resources in global scale (pervasive computing). The architecture was also conceived to incorporate the native adaptation mechanisms of technologies that compose it.

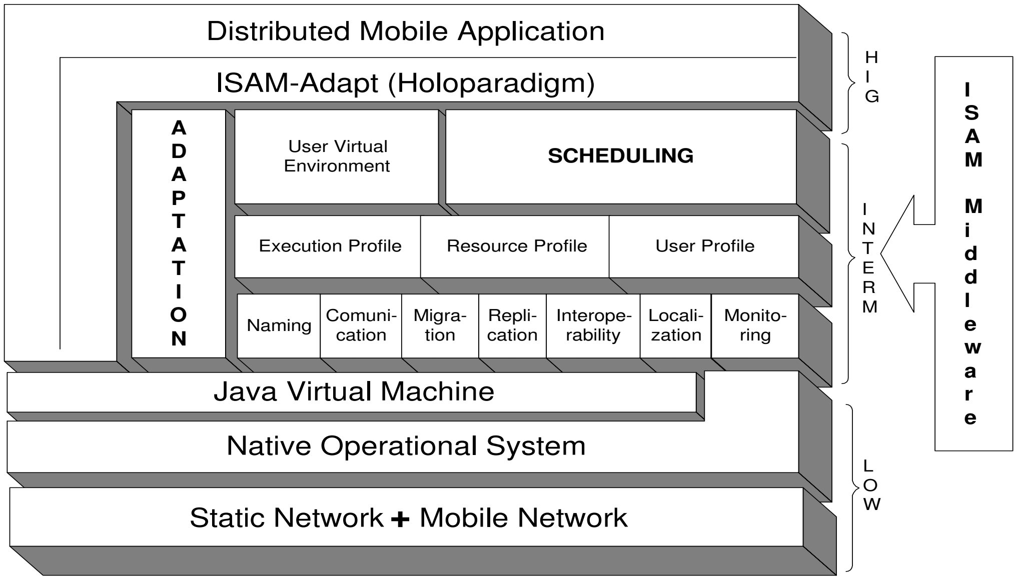

Figure 2: ISAM Architecture

The middle layer is the functional core of architecture, and it is provided with three abstraction levels. Two modules of application compose the first level: Scheduling and User Virtual Environment. The *scheduling* is the key component of adaptation, and some actions of the scheduler are described in Section 6. The *User Virtual Environment* is composed by elements that integrate the mobile user interface. The model was designed to support the exploration of *context-aware application* (adjust itself to resources, services and current location) and *personal applications* (adapt to interests and preferences of mobile user). The challenge of adaptability is to support users in different locations with different interaction systems that demand different presentation systems within the mobility limits. This module must characterize, select, and present the information according to the needs and context where the user is. To perform this task, the system is based on a usage model. It is formed by information, dynamically obtained, about user work, preferences, use patterns, physical movement pattern and user's hardware.

Therefore, the information that drives the adaptation decisions, from scheduler to applications, comes from three sources: execution profile, resource profile, application and user profile. The monitoring module obtains information by observing and registering both the application and the user behavior, every moment in every place. This information permits determining the historical and quantitative evolution of the monitored entities. The information interpretation determinates the user and application profile. So, the ISAM mobile applications can adapt to personal dimension as well as to temporal and spatial dimensions (also present in others systems). We are analyzing the possibility of using Bayesian Networks and Markov Chain for processing the information from environment monitoring. Some works considering four aspects of environment monitoring are being carried on: information about networking [26], dynamic information of application [2], static information of application [4] and information of the system itself [27].

On the third level of the middle layer are the basic services of ISAM execution environment that provide the functionality needs to the second level. It covers various aspects, such as: migration – mechanism that moves a component from one physical location to another one [6][27]; optimistic replication – mechanism that increases the availability and performance of the data access; monitoring – provides the information about the current execution context as well as application and user profile; location and naming – provide support for user movement among different *EXEHDAcells*.

4 Important Decisions for Architecture Design

In general, the adaptation is the process started as answer to an extern situation or event, and it results in the change of one element for another or in the change of service quality. The adaptation makes the system adjust itself to changes in resource availability in order to continue the work with the best possible quality, at each moment. Designing an adaptive system is a difficult task, which acquires more complexity when it comes to mobile computing. Researchers are started approaching this problem [1][5] [7][11][17][19][20][22]. The innovation of our approach is the uniform treatment of adaptation in all system, from base to programming paradigm, besides not being compromised to a specific application domain. These decisions are based on our belief that there is a huge potential in using mobile application in various application domains. So, it is necessary to provide an infrastructure for general-purpose applications, which permits to express the mobility and adaptability as simple as possible.

4.1 Communication and Programming Model

The upper layer of ISAM architecture (Figure 2) is composed by user application developed with the *HoloLanguage* [6], a programming language that combines logic, object-oriented and procedural paradigms. It uses a coordination model that supports implicit invocations (read and write operations in the blackboard are used to communication and synchronization) and explicit invocations (messages). The *HoloLanguage* supports yet concurrence and both physical and logical mobility; they are expressed in the semantic of the entity “being”. Figure 3 illustrates two possibilities of mobility, which are important for ISAM architecture: physical and logical mobility (A), and physical mobility (B).

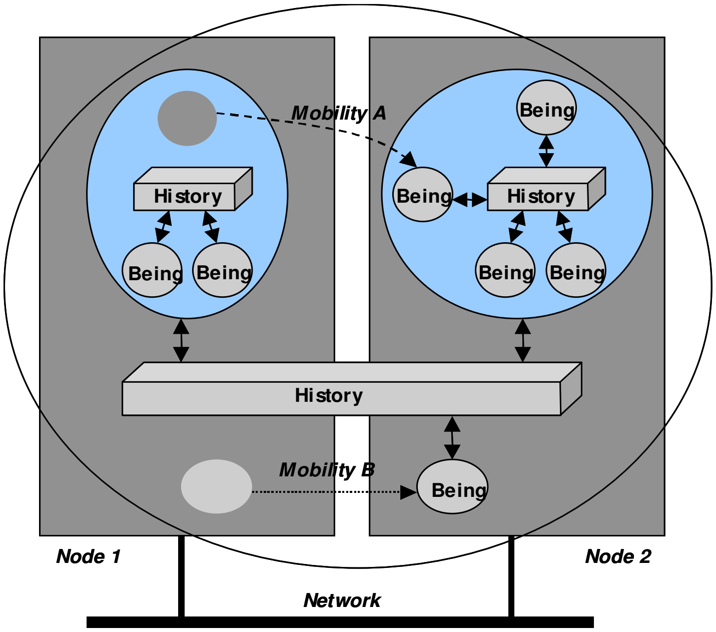

Figure 3: Mobility in *HoloParadigm*

In the *HoloLanguage,* the applications are modeled with “being” (existence entity) and symbols (information entity). There are two types of “beings” (Figure 3): elementary (atomic “being” without composition level) and composed (“being” composed by others “beings”). An elementary “being” is structured with three components: interface, behavior and history. The interface describes the relationships among “beings”. The behavior contains the actions that implement its functionality. The history is a shared storage space encapsulated in the “being”. The composed “being” has the same structure of the elementary “being”. However, it supports the co-existence of others “beings” in its composition (components “beings”). Each “being” has a history encapsulated inside the “being”. In case of composed “being”, the history is shared by component “beings”. The component “beings” participate in the development of shared history and they are submitted to reflexes of the changes in histories. So, it is possible the existence of several levels of encapsulated history. However, the “being” only accesses the history in its own composition level. The history is implemented as Multiple Distributed Tuple Space. At moment, the *HoloLanguage* is mapped to Java Language. This strategy allows exploring quickly the proposed language on prototypal applications.

The choice for the *HoloParadigm* (and its *HoloLanguage*) is motivated by the following questions. The mobile environment requires an interaction model suitable for reflecting the autonomy of involved components and that provides a spatial-temporal uncoupling [3]. Models based on communication by RPC (*Remote Procedure Call*) or ROI (*Remote Object Invocation*) are not appropriated, because they have communication mechanisms inherited synchronous, which cannot be guaranteed in the mobile environment. Interactions models based on messages are more flexible, especially considering the asynchronous messages. However, they can be subjected to the problem of scalability. Another issue is the identification of the final node that cannot be known in a highly dynamic environment. Nowadays, peer-to-peer models have called attention [8]. These models are suitable to mobile environment because they are inherently distributed, weakly coupled and scalable. The difficulty is introduced by highly variable network topology; it has its source in the movement of mobile hosts that compose the ad-hoc network. On the other hand, we consider that proposals of middleware based on Tuple Spaces [12][22] can be the answer for this question in the ISAM viewpoint. The “beings” interaction through the history implements this interaction model.

The highly heterogeneous systems, as the mobile, with very independent entities and parallel activities, require programming models that appropriately allow expressing the mobility. According to the ISAM proposal the adaptability must also be expressed. Traditional client-server model is the pull model, it is synchronous and with one thread. Therefore, it does not fill the requirements of a mobile environment. Alterations in this model can produce a push model, in which the server sends information to a receiver group that register the interest in a special event. Publish/subscribe systems do not request explicitly the task/information, and they can act as clients and servers simultaneously. This solution is event-based and deals with the requirements of uncoupled and asynchronous interaction, that are important to mobile environment. This model is also suitable for expressing the adaptive behavior of mobile application.

The mobility must be expressed by entities model with characteristics of autonomy, pro-activity, migration and reaction [3]. The “being” model is similar to mobile agents that present these characteristics [11][16]. In this sense, the application modeling with “beings” brings the advantage of joining two models: mobile agents and event-based models. It became attractive for expressing the mobility and the adaptability – ISAM Project goals.

4.2 Transparency x Awareness

There is one large difference between distributed system and mobile system. The first one explores the distribution transparency for the end user. Meanwhile, for the second one, a certain degree of mobility awareness (location, execution context) is important to adaptive behavior of application. Therefore, it is necessary to establish a tradeoff between awareness and transparency of mobility (in general, there is a conflict of interest between them). The system must provide information about the environment required by application for reacting to environmental changes. The current systems do not provide facilities for controlling the transparency degree required by mobile applications. Also in this aspect, ISAM architecture is different from others.

4.3 Adaptation Dimensions and Levels

Adaptation requires the existence of multiple ways of execution for an application, or alternate configurations, which exhibit different utilization profiles of computational elements. We observe that adaptation in mobile applications can occur basically in three dimensions: temporal, spatial, personal [3]. Temporal adaptation refers to "when" occurs the adaptation: at start-up (semi-static) and/or runtime (dynamic). Spatial adaptation is related to "who" suffers adaptation. Different from other adaptive systems, the mobility introduces the personal dimension: related to the behavior and preferences of the mobile user.

Besides, adaptation can occur in levels relating to adaptation focus: resource, context and situation. These levels represent the computational elements in which the application is self-interested, and influences its behavior or performance.

5 Towards a Taxonomy for Mobile Applications with Adaptive Behavior

We did not find out a proper taxonomy for mobile applications with adaptive behavior. Then, we propose a first classification, which could be source for future discussions. The taxonomy proposal is shown in Figure 4.

The bases of the classification are the answers to three questions: who executes the adaptation? who is subjected to the adaptation? how is the adaptation made?

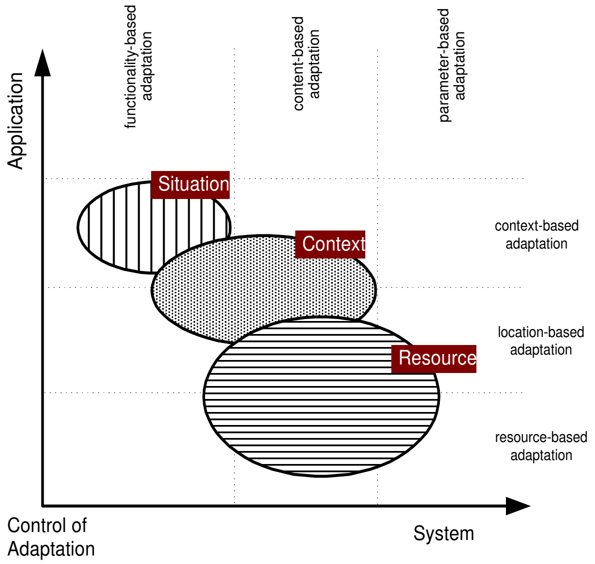

Figure 4: Mobile Applications Taxonomy

The responsibility for adaptation moves between two limits: system (automatic) and application (programming). To identify the application domain that, on one side, can take advantage of automatic adaptation and, on another side, must have interference of application or user, we have separated the elements that can be subject to adaptation: resource, location and context. Resource includes physical and logical computational elements, related to network, mobile and static device, files, programs and others. Location is the physical position of user with his mobile device. The context is a wide concept. In ISAM Project context is defined as "all relevant information to the application that can be used to define its behavior" [3]. In a preliminary analysis, the context is determined by information of whom, when, where, what is making and with what is making it. This information can be obtained in the mobile part and in the static part of the environment by monitoring module of architecture.

The reaction to changes in availability of any of these elements can be implemented using various techniques and mechanisms. In general, parameter changes, content or format changes and functionality changes are used for that.

The combination of these three aspects shows us the features of many domains in mobile applications, as exemplified below.

The *network-aware applications* adapt to changes in availability of network components, specially the bandwidth. In many of these applications [1][5][7], the adaptation is related to data reduction, transformation, filtering techniques for network transport. In these systems, the component locations are static; the adaptation is raised for network usage change. The network can be viewed as a resource; therefore, this kind of application is a subset of the resource-aware application.

The *resource-aware applications* [19][22] try to identify and to use the available local resources. It adapts to provide an appearance of continuous service, and to manage the tradeoff among transparent, flexible and efficient use. Most of the adaptation process can be executed by the system (automatically) using parameter adjustments (simples cases), or format changes of components. For example, in multimedia domain it is common to reduce the color numbers or frame numbers in image for adjusting to bandwidth reduction. Another example, the unavailability of database required could force the application to migrate to another site where it is easy to get to it [16].

The *location-aware applications* [26] have their behavior determined by the physical location of the user. They use the knowledge of location as background for replying to more specialized queries, or for executing a location-dependent action, or yet for providing information about the area and the objects around. Here, there is an effective collaboration between system and application for adaptation, probably using strategies of content adjustment. This is a context-aware application subset.

The *context-aware applications* [10][13] can deduce its internal and external state using sensors or monitors, and alter their functionality according to the new state. These applications must be personalized by the user or by his behavior, based on where they are being used. The context describes information related to: the user (capabilities and preference exchange, ad-hoc preferences), location (logical, domain), devices, hardware profiles, network (bandwidth, disconnection rate, price) and computational objects (availability, reliability, security). The access to this information allows the application/system to choose the best options among alternative adaptation actions.

The most general kind of application is the one that is aware of other applications around and their context of usage: *situation-aware application*. The adaptation occurs in functionality changes depending on usage context and user preferences. One example is when the mobile user is at home, reading the newspaper in his mobile device. When he moves to his car, he wishes to change automatically the application looking for the weather conditions or travel information.

This taxonomy answers to questions on "who" and "how". Another two questions "when" and "where" depend on the implementation of support to adaptation. The first, when occurs the adaptation, is related to concept of system agility [21]. The second, where it occurs depends on the characteristics of the system equipments. Depending on the mobile equipment capacities, the adaptation mechanisms can be more flexible and act in both, mobile and static hosts.

6 Multilevel Collaborative Adaptation

One important design goal is making the whole system as adaptable as possible for attending the needs of the mobile applications. For that, ISAM uses a *multilevel collaborative model* in which both the system and the application contribute to adaptation at load-time and at run-time. The scheduling process – the core of ISAM architecture, manages this collaboration.

So, adaptation is not fully transparent to the application. Applications can control this process, depending on the application domain and its purpose. For that, the application inserts commands of *HoloLanguage* [6]. The language commands express the desire of adaptation and drive the scheduler in his adaptation decisions. As viewed in Section 3, the information that guides the ISAM adaptation decisions comes from three sources: execution profile, resource profile, application and user profile. This way, applications can have an adaptive behavior on three dimensions: temporal, spatial and personal. Besides this, the application can be susceptible to abstract elements, such as location, resource, context or situation.

In Multilevel Collaborative Adaptation model:

- the system is responsible for some adaptations related to performance or resource management;
- the application is responsible for decisions of adaptations related to situation of usage or specificity of its domain;
- both are responsible for adaptation decisions negotiated in many cases.

The problem is (a) to determine the cooperation degree required by system and application for each application/domain and (b) to create protocols acceptable for both. So, the proposed taxonomy (Section 5) is a first result for getting the answer of this issue. Unlike other systems, where adaptation usually takes place at compile or build time, the changes to mobile systems must take place more dynamically while the system is actually being used. Those changes will occur automatically when possible and with collaboration of the application when necessary.

In many cases, the decision of collaboration is started by application, depending on the purpose of that particular application. The flexibility is achieved by language abstractions and by corresponding runtime support. The language elements work independently, or it cooperates with the system to achieve the desired level of adaptability at runtime.

In pervasive computing environment, locations have a profound effect. As a user moves physically his applications need to be continuously available. He can carry them in his mobile device. But this can be impossible due to limited capacities of mobile devices. So, the applications must reside on static hosts and to be sent to mobile user by demand. This is a main function of *User Virtual Environment* (UVE) module of architecture that operates together to the scheduler (Figure 2). The collaborative-adaptive behavior of system (scheduler and application) is broad in possibilities. To illustrate it, we discuss some possibilities of ISAM adaptation in two moments: at load-time and at run-time.

6.1 Load-time Collaborative-Adaptation

ISAM maintains information about user and his applications on UVE. Information includes user identification (login, password, *HoloHome*), last application executed, last contact point to network (*EXEHDANode*), user profile and user preferences. When the user connects to network, the scheduler sends him the last UVE stored, after making some adaptations. This adaptation is made based on capacities of current user's mobile device, connection qualification, location, profile user and application. According to the possibility, it is sent the complete UVE or only basic components. The scheduler finds the last UVE in the last contact point of user to network (*EXEHDANode*). If it does not locate the UVE, it searches in the user's *EXEHDABase*; which acts as a backup for user's information of the system. When the connection quality increases, the scheduler can make a prefetching of others components according to the user and the application profile.

Another example of adaptive behavior is related to the code. The application code is partitioned. Each "being" has its own code, stored in a file. The "being" has a header that contains loading information for the scheduler. One of this piece of information refers whether the code that must be sent to a mobile host or to a static host, and if it must be allocated near the resources (co-allocation). This way, the scheduler can decide where to create a “being”. The piece of information that drives the scheduler decisions is generated by instructions of *HoloLanguage* (ISAM Adapt).

6.2 Run-time Collaborative-Adaptation

Later, when the systems state changes, the scheduler uses context information to dynamically reallocate, reschedule and restructure the applications. The scheduler automatically makes some adaptations, such as migration and database replication started by load-balancing. The “being” code load is also dynamic. The scheduler searches for the nearest “being” code of the *EXEHDANode* which contains it.

To react in context changes, the system needs information obtained on the monitoring module (Section 3). The monitoring is independent; it must send information to many applications and to the scheduler. The application registers its interest on state changes of some resources [3]. For that, the system offers a resource model [24] that allows the programmer to (re)define the attributes and the monitor code of elements that affect the application behavior, and whose state changes the application desires to be notified for adapt it [3].

7 Related Work

Noble [20] has inspired our work. According to him, the adaptation strategies are between two extremes: responsibility of application (*laissez-faire*) and responsibility of system (application-transparent). In his thesis, he proposes the application-aware adaptation that supports a collaborative relationship between application and system. However, that prototypal system is only based on network-aware application. Many other systems that provide an adaptive behavior to mobile applications were also analyzed, among them [5][7][10][11][13][16][17][18][22]. In general, these systems provide only the resource level adaptation. Its applications domains are specific, such as multimedia [19] and location-depend information [26]. The adaptation strategies are simple, such as data format-oriented adaptation [19], data replication [5] and migration thread [22]. In contrast to ISAM, those systems cover a little range of adaptation possibilities, which are limited for expressing the flexibility need to new types of applications that are context-aware.

More similar to the mobile environment viewer of ISAM Project is Grimm’s. His work, One.world [17], also raises the question on how to structure systems that can support pervasive applications. As ISAM, Grimm states “the system cannot automatically decide how to react to change, because there are too many alternatives.” One.world introduces a system architecture based on three principles: exposition of changes, dynamic composition of applications and services, separation data and functionality. Applications store and communicate data in the form of *tuples* and are composed by components, which implement its functionality and interact by importing and exporting event handlers. Environments (serving as containers for tuples, components and other environments) provide structure and control. This are similar to *HoloParadigm´*s composed “being.” As ISAM, One.world uses the Java language to provide a uniform and safe execution platform. However, a significant difference between ISAM and One.world is related to resource changes. One.world requires that all resources should be explicitly bound and leased, while ISAM supports dynamic adjust to new state based on system-application negotiating. Furthermore, the ISAM architecture is wider than One.world, which supports only services for mobile computations using the class loader of Java, persistent storage via tuple spaces implemented in T Space, and resources discovery implemented using Jini service.

8 Conclusions

Mobility is fast becoming a key feature in software development. Mobility support brings a set of new requirements and challenges to software production, and creates demand by new applications types, named location-aware applications, context-aware applications, and pervasive applications. The core of these challenges is a dynamic operating environment, started by users movement from different terminals, locations and contexts and by own nature of mobile environment. The ISAM Project, differently from other systems, has as objective to deal with the adaptation of a uniform mode providing three dimensions: temporal, spatial and personal. Another key point is the collaborative behavior of adaptation. The system, as well as the application collaborates to adaptation decision at load-time and at run-time. This approach allows more flexibility in the modeling of the distributed mobile applications. We argue that to simplify the application implementation is necessary a programming style that expresses mobility and adaptability on a natural way. The runtime language must provide mechanisms for specialized adaptive behavior. Furthermore, the support system must be modeled under the optic of general-purpose. Also, it must have a language to implement general-purpose application. This way, it can attend the expressiveness requirements of several applications. Our proposal architecture is broad and presents many challenges. At moment, several researches are ongoing, which approach various issues, such as: language, runtime, monitoring and integration with current technologies.

References

[1] O. Angin "The Mobiware Toolkit: Programmable Support for Adaptive Mobile Networking", *IEEE Personal Communications Magazine - Special Issue on Adapting to Network and Client Variability*, aug., 1998.

[2] E. B. Araujo, I. Augustin, A. Yamin; L. Silva, C. Geyer, "A Monitoring Proposal for Visualization of Distributed Applications Java", *Chilean Congress of Computing 2001 - V Workshop Distributed and Parallel System*, Chile, nov., 2001.

[3] R. A. Real, A. Yamin, I. Augustin, J. Barbosa, C. Geyer. "Tratamento da Incerteza no Escalonamento de Recursos em Pervasive Computing". *IADIS International Conference on WWW/Internet*, Lisboa, IADIS Press, p. 167-170, 2003

[4] S. Azevedo, P. Vargas, J. Barbosa, A. Yamin. C. Geyer, "DEPAnalyzer: An Static Analyzer of Dependencies in Java Programs", *Workshop of High Performance Computational System (II WSCAD)*, Goiás, Brazil, sep., 2001.

[5] A. Baggio, "System Support for Transparency and Network-aware Adaptation in Mobile Environments", *ACM Symposium on Applied Computing - Special Track on Mobile Computing Systems and Applications*, Atlanta, USA, feb., 1998.

[6] J. Barbosa, A. Yamin; P. Vargas; I. Augustin; C. Geyer, "Holoparadigm: a Multiparadigm Model Oriented to Development of Distributed Systems", *International Conference on Parallel and Distributed Systems*, New York, IEEE Press, p.165-170, 2002.

[7] J. Bolliger, T. Gross, "A Framework-based Approach to the Development of Network-aware Applications", IEEE *Transactions on Software Engineering*, v.24, 1998.

[8] P. Charas, Peer-to-Peer Mobile Network Architecture, 1st IEEE *International Conference on Peer-to-Peer Computing*, Sweden, aug., 2001.

[9] L. Capra, W. Emmerich, C. Mascolo, "Reflective Middleware Solutions for Context-Aware Applications", *UCL-CS Research*, University College London, mar., 2001.

[10] G. Chen, D. Kotz, "A Survey of Context-aware Mobile Computing Research", *Technical Report TR2000-381*, Dartmouth Computer Science, 2000.

[11] A. Corradi, P. Bellavista, C. Stefanelli, "Mobile Agent Middleware for Mobile Computing", *IEEE Computer*, mar., 2001.

[12] N. Davies, A. Friday, S. Wade, G. Blair, “Limbo: a Tuple Space Based Platform for Adaptive Mobile Applications”, Proc. *International Conference on Open Distributed Processing / Distributed Platforms (ICODP/ICDP)*, Canada, may, 1997.

[13] N. Davies, “The Design of an Object Model for a Context-Sensitive Tourist Guide”, Proc. *Interactive Applications of Mobile Computing*, Germany, 1998.

[14] W. Emmerich, “Software Engineering and Middleware: a Roadmap”, In: The Future of Software Engineering. 22nd *International Conference on Software Engineering*, ACM Press, may, 2000.

[15] K. Geihs, “Middleware Challenges Ahead”, *IEEE Computer*, jun., 2001.

[16] R. Gray, W. Caripe, G. Cybenko, K. Moizumi, ”Network-Awareness and Mobile Agent Systems”, *IEEE Communications Magazine*, Jul., 1998.

[17] R. Grimm, “Programming for Pervasive Computing Environment”, 18th *ACM Symposium on Operating System Principles*, Canada, oct., 2001.

[18] A. Harter, “The Anatomy of a Context-aware Application”, Proc. 5th *International Conference on Mobile Computing and Networking (MOBICOM´99)*, Seattle, USA, aug., 1999.

[19] T. Kunz, J.P. Black, An Architecture for Adaptive Mobile Applications. Proc. 11th International Conference on Wireless Communications, Canada, jul., 1999.

[20] B. Noble, System Support for Mobile, Adaptive Applications. *IEEE Personal Computer Systems*, 7(1), feb., 2000.

[21] B. Noble, M. Satyanarayanan, D. Narayanan, J.E. Tilton, J. Flinn, K.R. Walker, “Agile Application-aware Adaptation for Mobility”, Proc. *16th ACM Symposium on Operating Systems Principles*, France, oct., 1997.

[22] G.P. Picco, A. L. Murphy, R. Gruia-Catalin, “LIME: Linda Meets Mobility,” Proc. *21ST International Conference on Software Engineering* (ICSE´99), Los Angeles, USA, may, 1999.

[23] M. Tanganathan, A. Acharya, J. Saltz, “Sumatra: a Language for Resource-aware Mobile Programs”, In *Mobile Objects Systems: Towards the Programmable Internet*, Springer-Verlag Publisher, Series Lecture Notes on Computer Science v.1222, apr., 1997.

[24] C. L. Reis, “The APSEE Modeling to Resource Modeling and Management in Software Process Environment”, *Brazilian Symposium on Software Engineering*, Rio de Janeiro, Oct., 2001.

[25] M. Satyanarayanan, “Fundamental Challenges in Mobile Computing”, Proc. *15th ACM Symposium on Principles of Distributed Computing*, 1996.

[26] E. N. Silva Jr, I. Augustin, A. Yamin,; J. Barbosa; C. Geyer, “Network Management Hierarchy with Mobile Components”, *VIII CACIC Argentine Congress in Computing Science*, Argentina, oct., 2001.

[27] L. Silva, I. Augustin, A. Yamin, J. Barbosa, C. Geyer, “Mechanism to Support Scheduling for System with Java Distributed Objects”, *VIII CACIC Argentine Congress in Computing Science*, Argentina, oct., 2001.

[28] R. Want, B. Schilit, *IEEE COMPUTER. Special Issue on Location-aware Computing*, 34(8), aug., 2001.

[29] A. Yamin, I. Augustin, J. Barbosa, C. Geyer, “Exploring the Scheduling in Mobile Distributed Applications Performance, *Workshop in High Performance Computational Systems*, Goias, Brazil, sep., 2001.