

Towards a Context-Oriented Services Discovery and Composition Framework

Soraya Kouadri Mostéfaoui
Department of Computer science
University of Fribourg
Chemin du Musée 03 FR-1700
Switzerland
Soraya.kouadrimostefaoui@unifr.ch

Abstract

The development of new services through the integration of existing ones -referred to as service composition- is generating considerable interest in several computer science communities. Existing service composition systems miss out on the opportunity of enormous benefits by exploiting useful contextual information relevant to service discovery and composition [Lee and Helal, 2003]. This paper presents our initial research efforts on the CB-SeC framework (CB-SeC stands from Context Based Service Composition). The approach adopted in this framework is the combination of context-aware computing and agent-oriented computing, in order to provide better user-tailored services in pervasive environments. Agents have the advantage of being able to assist users, to discover and compose services. Agents are also mobile and are able to deploy functions on the fly [Khedr et al, 2002]. This helps with fast and autonomous adaptation to context changes.

1 Introduction

Weiser's [Weiser, 1991] vision of ubiquitous computing, with its invisible yet attentive computing environment, providing the right information to the right person at the right time, is an exciting vision of how to evolve computer technology [Prekop and Brumett, 2002]. Until now considerable work has been done in creating the physical computing devices needed to fulfill Weiser's [Weiser, 1991] vision, in contrast little work has been done in devising ways to make the devices and services smarter, more sensitive, and aware of their user, and their user's environments [Prekop and Brumett, 2002]. Context and context-awareness may provide a ubiquitous environment with the ability to adapt the services or information it provides by implicitly deriving user's needs from the context that surrounds the user. By context, we refer to the circumstances or situation in which a computing task takes place.

This paper presents our initial research efforts in the CB-SeC framework (CB-SeC stands from Context-Based Service

Composition). The approach adopted in this framework is the combination of agent-oriented computing and context-aware computing in order to provide better user-tailored services in ubiquitous environments [Kouadri et al, 2003]. Unlike other systems that miss out on the opportunity of enormous benefits by exploiting useful contextual information [Lee and Helal, 2003], the approach adopted in our framework is able to capture many aspects of context; to enable much more sophisticated discovery and composition of services. The context awareness we have promoted considers the user's context, the computing context, the time context, and the context history.

The rest of this paper is organized as follows. Recent developments in wireless communication, Internet services and agent technologies are presented in section 2. A motivating scenario is given in Section 3 to illustrate how and what context information could be used for dynamic service discovery and composition. Section 4 focuses on the context modelisation in CB-SeC. In Section 5, the main components of the CB-SeC framework are introduced. Sections 6 and 7 describe the context layer and the middleware services layer respectively. Finally Section 8 draws the conclusions.

2. Enabling Technologies

This section presents the recent developments in wireless communication, Internet services and the agent technologies.

2.1 Wireless Communication

With the proliferation of mobile computing devices such as laptops, mobile phones, personal digital assistants (PDAs), and wearable computers. More companies are offering services to users over wireless devices [Maamar, et al, 2002]. Reading emails using mobile phones, and surfing on the web thanks to the Wireless Application Protocol, WAP is becoming more and more popular. WAP is both a communication protocol and an application environment. As a communication protocol it is designed to work with most wireless networks such as CDPD, CDMA, GPR. As application environment it can be built on any operating system

including PalmOS, Windows, and JavaOS. Other emerging communication standards like Bluetooth¹ are among the other evidences of the wireless technology expansion.

2.2 Internet Services

There are several industrial initiatives to enable "web services" which integrates PCs, mobile devices, databases, and networks into one virtual fabric, that users could work with via browsers [Bhaskaran et al, 2002] [Vaughan, 2002]. These include HP's e-speak and the web services platform, Microsoft's .NET, and Sun ONE. These systems are based on a language called WSDL² (Web Service Description Language) for description of Internet services, a common format for these descriptions the SOAP³ protocol, and registry to support service location the UDDI⁴ (Universal Description, Discovery and Integration of web services).

2.3 Agent Technologies

Agents can simply be defined⁵ as software components that are capable of autonomous actions in some environment in order to meet their design objectives [Jennings and Wooldridge, 1998] [O'Hare and O'Grady, 2003]. Roughly agents can be classified in two classes: weak agents that respond on a stimulus response manner and minimize the need for complex representation model of the agent environment. And strong agents that maintain mental states that are used to support rational reasoning in a collaborative context. Within the mobile computing area the use of agents is beginning to find flavor although many of the agents utilized are of the weak variety, due to the computational restrictions of the mobile devices (memory, processing speed, screen size...etc). However recent advances both in the emergence of agent prototyping environments for developing Java agents (Aglets⁶, Jack⁷, JATlite⁸), and the availability of a micro edition of Java that can be hosted on all PDAs, namely the J2ME⁹ (Java 2 Micro Edition), have meant that stronger notions of the agenthood can now be used in the mobile and ubiquitous sector [O'Hare and O'Grady, 2003].

3 A Motivating Scenario

This section presents a simple motivating scenario, to illustrate the merits of the application of context-awareness in the service discovery and composition process. Suppose the situation where a German-speaking student at the Unifr campus is planning to go out Friday evening. She wants to eat in a restaurant. She wants to have the bus schedule in order to know which bus to take. Finally she wants to print all these information. This

situation requires at least three services: restaurant searching, bus schedule, and a printer service, to be connected and executed according to a specific order. Using a classical service discovery and composition process and without using any contextual information; services will be matched against her query, composed, and the resulting set of composite services will be sent to her. For example a big list of restaurants descriptions will be sent to her device, and she will manually browse the list to figure out which restaurant she wants. Unfortunately, it may take a long time to find manually the one that satisfies all her requirements (e.g. prefers vegetarian food...etc). It may also take a long time for here to manually translate the restaurants descriptions to German. However using the contextual information when discovering services, she will gets back a small restaurant list stored in the order of her preferences, and the nearest one will be recommended through the evaluation of her location attribute (e.g. ZIP code). Restaurants' descriptions may be translated and sent in German rather than in the default language set by the service provider. Knowing that she is a German speaking person (from her context profile), a language translation service (e.g. the Babelfish service) is composed-on behind the scene in order to provide her with a better-tailored service.

As a result mobile users may see a different set of composite services even with the same query [Lee and Helal, 2003], e.g. in case of a Japanese speaking person the Babelfish service is composed-on with the restaurant searching service in order to translate information in Japanese, another service is also needed, and assembled (composed-on) with the two previous ones, for reformatting for Japanese style information appliance displays.

After choosing the restaurant she wants to print the bus schedule. Without using any contextual information she will have back a list of all the buses of the day printed on the default printer, and she will manually looks for the one she can take. However knowing the current time, the day of the week, and her current location; only buses that run for the rest of the day will be printed on the nearest printer (according to her location: building, room...etc), rather than on the default printer that may for example be far from her current location.

4 Modeling Context in CB-SeC

This section presents our modeling concepts designed to capture many of the features of context information that are relevant to the service discovery and composition process.

4.1 Required Context Types in CB-SeC

Composed of *con* (with) and *text*, context refers to the meaning that must be inferred from the adjacent text. Many authors such as Schilit [Schilit and Theimer, 1994] and Ward [Ward, et al, 1997] tried to map this definition to the mobile computing world, however all their definitions lack the generality. Dey [Dey et al, 2001] has reviewed these defini-

¹ <http://www.bluetooth.com>

² <http://www.w3.org/TR/wsdl>

³ <http://www.w3.org/TR/SOAP/>

⁴ <http://www.uddi.org>

⁵ Note that this is not a formal definition.

⁶ <http://www.trl.ibm.co.jp/aglets>

⁷ <http://www.agent-software.com/shared/demosNdocs/jack/html/>

⁸ <http://www-cdr.stanford.edu/ProcessLink/papers/JATL.html>

⁹ <http://java.sun.com/j2me/>

tions, and defines context as: "Any information that can be used to characterize the situation of an entity, where an entity can be a person, a place, a physical or a computational object". Given this definition, the amount of information that can be included in context is very vast [Khedr et al, 2002]. However nowadays, for most of the context-aware applications, the usual context parameter is location. As part of our research we needed to clarify the types of context required in a service discovery and composition process. We identified five types of context [Kouadri, 2003].

1. *User context*: role, identity, location, preferences, social situation, permission profile...etc
2. *Computing context*: network connectivity and nearby resources (printers, displays, and workstations) ...etc.
3. *Time context*: time of a day, week, month, and seasons of the year...etc.
4. *Physical context*: such as weather, temperature...etc.
5. *Context history*: more importantly, when the user, computing, and physical contexts are recorded across a time span, we obtain a *context history* [Chen and Kotz, 2000], which is also useful when discovering and composing services.

4.2 Abstract Representation of Context Information

To proliferate context information in a timely due manner and general format, a representation of context information based on the resource description framework (RDF) [Lassila and Swick, 1999], called the CSCP (*Comprehensive Structured Context Profiles*) is used. First introduced by A. Held and al. in [Held et al, 2002], this representation expresses information by means of session profiles. A session profile describes all relevant context information of the session, in our case a session corresponds to the request taken by the Context Broker Agent. The CSCP is an RDF based meta language. As a descendant of RDF, CSCP inherits the interchangeability, decomposability and extensibility of RDF. CSCP interchangeability is based on the XML serialization syntax of RDF [Held et al, 2002]. Furthermore, CSCP provides features to attach conditions and priorities to attributes (Fig.1). This extends the means to express user preferences. By assigning conditions to user preference attributes, by means of multiple conditional RDF statements about an attribute, If-Then-Else-If expressions may be formulated.

5 The Conceptual Framework

In this section, we describe a general layered architecture that enables a context-based service discovery and composition in pervasive computing environments.

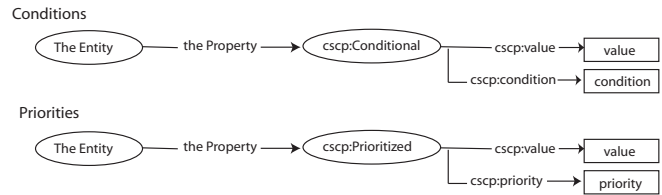


Fig. 1. RDF graph notation of CSCP conditions and priorities

4.1 The CB-SeC Architecture

The architecture of the CB-SeC framework is depicted in (Fig.2). CB-SeC is composed of four core layers; each of them will be described in turn.

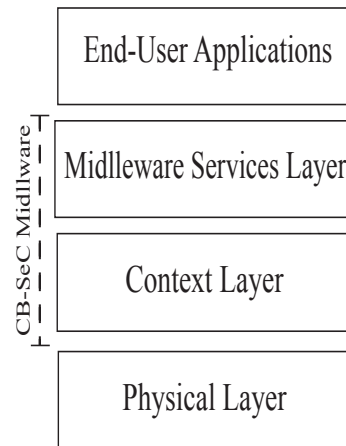


Fig. 2. The CB-SeC Architecture

The Physical layer: represents the resources belonging to the environment. Resources describe all the entities that may be involved in the execution process of an application; so they describe physical devices, software components and sensors. Due to the dynamic changes in the availability of physical devices, network bandwidth, connectivity and user location, the system has to classify those resources monitoring constantly their changes. In such a way the upper layers have always a consistent description of the physical dimension. [Kouadri et al, 2003].

The Context layer: this layer is responsible for gathering and processing contextual information. It consists of two modules: the Context Gathering Engine (CGE), and the Context Data Base (CDB). We will come back to this layer in greater detail when in section 6.

The Middleware Services layer: this layer is the core part of our framework; it is responsible for carrying out the process of discovery, composition and execution of services. It is in turn composed of four modules, the Context-Based Service Discovery module, the Context-Based Service Composition module, the Service Execution module, and the Cache Engine. These modules are tightly

coupled with each other due to their inter-dependencies; each of them will be described in details in section 7.

The End-User Application layer: is a generalization of the view in a traditional MVC; it works as a presentation module embodying any software that utilizes our context-based service composition framework. It encompasses different GUI facilities with whom users can customize their applications by setting their preferences, permission profiles, i.e. which service can connect the user and when. In order to provide more personalized and helpful composite services [Kouadri et al, 2003].

6 The Context Layer

We have identified two main operations necessary for context awareness in the CB-SeC system, namely context acquisition and context representation, for each of these components we have associated a computational module, respectively, the Context Gathering Engine and the Context Data Base, (Fig.3) shows the architecture of the context layer.

6.1 The Context Gathering Engine (CGE)

The Context Gathering Engine (CGE) is responsible for gathering, processing and interpreting the users contextual information. It consists of a Multi-Agent System (MAS), in which each agent is associated with one type of contextual information. The agents are responsible to gather the data provided by the software and hardware sensors, to interpret the gathered raw sensor data into meaningful information and finally to extract the specifications. This does not mean that all the contextual data must be transferred via the wireless link. It may rather contain references to external resources, such as the device defaults that can be retrieved from the device vendor's web site.

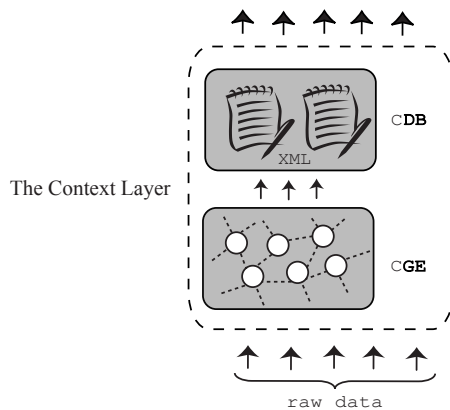


Fig. 3. The Context Layer

6.2 The Context Data Base (CDB)

The context database is acting as a bridge repository; it receives the specifications of the contextual information for each user from the agents of the CGE. These specifications are translated into XML documents for subse-

quent use by the Context-Based Service Discovery Module of the above layer. An important part of the context database is the classification of the context information into context types, e.g. both the GPS and the network-based locator provide location information. The specification of context type is achieved using an XML template that defines the kind of information this type offers. Using this approach, the Middleware Services Layer can retrieve the specific contextual data in a way that is decoupled from the service used for acquiring the data. This method of hiding the actual mechanism for retrieving contextual information, allows the CB-SeC framework to coordinate the access to context for different applications [Kouadri et al, 2003].

7 The Middleware Services Layer

The Middleware Services Layer (Fig.4) is responsible for the process of carrying out the discovery, composition and execution of services. It is composed of four modules: the Context-Based Service Discovery Module, the Context-Based Service Composition Module, the Service Execution Module, and the Cache. Each of these modules will be described in turn after introducing how services are modeled in CB-SeC.

7.1 Service Modelisation

The service modelisation (description) is a meta-level representation of the actual real world service; it describes the behavior of the service, and can be regarded as a system specification in the sense used in [Jackson, 1995]. Thus, it serves as the bridge between the real world and the device. The discovery of a service involves the discovery of its description that can be used to access the service [Lee and Helal, 2003]. In this sub section we take a look at how CB-SeC describes its services.

Each service description must contain three parts: the *interface*, the *capsule*, and the *constraints description*.

1. The interface: contains information regarding the operations that can be invoked in a service and their respective input and output parameters.

2. The capsule: gives information on where the service is located (URL) and how it can be accessed, the invocation protocol(s) (HTTP: HTTPS, SMTP...etc), the types(s) and the port(s).

3. Constraints description: it is often necessary to describe the constraints regarding the parameters of a service, it may exist two service implementations having the same interface description but having different constraint descriptions [Kouadri et al, 2003] [Lee and Helal, 2003].

7.2 The Context-Based Service Discovery Module (CSD)

This module is required for the proper functioning of the service composition module. An important component of

the CSD is the *Context Brokering Agent*, (CBA) that functions is to match and recommend appropriate services among a possibly large set of service instances, taking benefits from context-information.

When a client requests a service it may be necessary to compose a complex service out of the registered basic services. The current implementation of our architecture uses UCM's social laws (UCM stands from Ubiquitous Coordination Model) to decompose complex service requests into basic services, and to determine the process model of execution. The UCM model is an instantiation of the unified coordination model XCM presented in [Courant et al, 2003]. It is viewed as an organizational domain composed by autonomous entities. An entity is defined by its structure obtained by a recursive composition of entities. Inside this domain, entities interact with each other through communication endpoints that define a set of actions; social-rules are defined to restrict the composition of basic communication actions. More details about UCM are available in [Courant et al, 2003].

Clients request services in a context describing characteristics, and time of the requested service, furthermore context contains information about the client and its preferences. The CBA's first job is to extract the contextual information from the Context-Data Base, then using these information it tries to discover the basic services against the ones registered on its vicinity (using the location attribute), it first matches the user's query as well as the user's device capabilities (using the computing context) into the nearby service registry and progressively increases its search radius [Chakraborty et al, 2002] to discover all the different services necessary for the composition. The CBA has to figure out other-end requirement, i.e. device capabilities anticipated by service implementations. It can get such information from the constraints field shipped with the services.

The previous step produces a set of candidate services, the CBA is then responsible for iterating through this set, and for picking the best offers according to the current context and the constraints set by the service providers. In other words services are incrementally filtered and ranked [Lee and Helal, 2003] according to the evaluation of the context parameters, to ensure that clients are given the best service instances.

7.3 The Context-Based Service Composition Module (CSC)

This module is responsible for carrying out the process on managing the discovery of services to yield a composite service. It further refines the set of selected services using the remaining context parameters that are relevant for the composition rather than for the discovery. Example of such a parameter can be the user preferences that are applicable to the overall composite service e.g. a person planning for her summer vacation wont spend more than 2000CHF for the whole vacation composite service i.e. (hotel service, car rental service, attractions, plane tickets ...etc), another ex-

ample may be the permission profile set by the user (e.g. when in class between 8h00 and 12h00 I do not want to receive SMS messages). The composite service that satisfies all the user preferences is chosen and the corresponding capsules (i.e. information about the services, addresses, ports, and invocation protocols) are sent to the Service Execution Module. In the mean time a copy of the process model as well as its context is kept in cache.

7.4 The Service Execution Module (SE)

This module is responsible for carrying out the execution of the composite service. Prior to this the Context-Based Service Composition module provides a feasible order in which these services can be executed. Mainly two types of execution exist according to the nature of the service, which can be either an electronic service (e-service) or a mobile service (m-service) [Maamar, et al, 2002]. For m-services execution is always local to the device of execution since the service it self has been transported to the device. However for e-services the execution can be remote or local. In the remote invocation the client sends remotely a request to a provider asking the execution of a service. The execution takes place in the provider platform. In local invocation the client asks remotely to transfer a copy of the service to its site. After being transferred, the execution takes place in the client site [Maamar, et al, 2002] [Kouadri et al, 2003].

7.5 The Cache Engine (CE)

The principle of the Cache Engine is to store expensive (time-consuming) information to create, so it can be automatically reused. For example if a composite service requires a complex calculation, caching may save processing time, speed up the response rate for the client and lessening the burden on the service server's CPU. In other words the Caching Engine stores the resulting process model of a query as well as its context, and reuses it for other clients that supply the same query with the same parameters. Caching is also useful when the same client supplies the same query in other contexts; first the cache is checked if the same composite service would satisfies the new query needs (according to the new context), if not the composite service is decomposed, services that do not meet the new requirements are composed-out, and a new discovery/composition process is triggered for the composed-out services. The new discovered services are composed-in with the cached ones. Finally the resulting composite-service is recommended to the user, and a copy of the process model is kept in cache as well as its context. Another advantage of the Caching Engine is to provide the system with a fault tolerance mechanism. We are aware that maintaining stable communication channels during the whole discovery/composition and execution process may not be possible in a highly dynamic and pervasive environment. Breakpoints are introduced in the execution process, and the broker for a particular request sends back the results after a sub-task is completed, these results are cached. When a failure occurs instead of mak-

ing the CBA initiating a new discovery/composition request, it reconstructs only the query that is still unsolved.

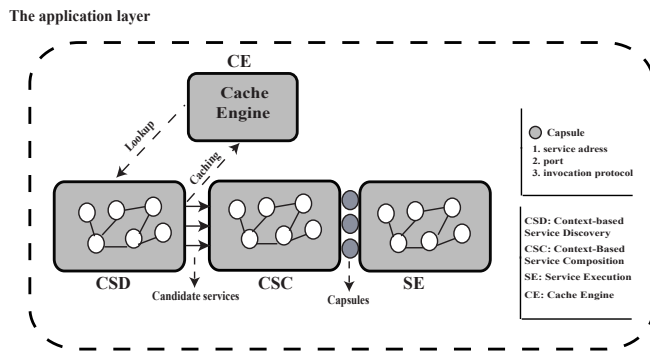


Fig. 4. The Middleware Services Layer

8 Conclusions And Future Works

Our ongoing and future work covers several research thrusts. Security thrust is one of them. Since the information stored in the user profile is held on behalf of the user, it is essential that she/he controls the authorization given to other applications and services. It is also very important to enable users to switch off access to their context information to everyone or to some services and applications. All these security aspects are being studied within the CB-DA project [Kouadri and Brézillon, 2003] (A Generic Framework for Context-Based Distributed Authorizations) at the Software Engineering Research Group at the University of Fribourg in Switzerland. Our second research thrust concerns the extension of the system with a Dynamic Context-Based Broker selection technique, in which the broker selection depends on the current context, the power of the platform, its stability, the number of nearby services...etc. Furthermore, work continues on representing and recognizing context. In the mean time and in order to have a first verification of the model two projects¹⁰ based on CB-SeC are under the way, emphasizing the use of the framework.

References

[Weiser, 1991] M. Weiser. *The computer of the 21st century*, Scientific American 1991.

[Prekop and Brumett, 2002] P. Prekop, and M. Brumett. *Activities, Context and Ubiquitous Computing*. Computer Communications, special Issue on Ubiquitous Computing. 2002.

¹⁰ One is a Master thesis that explores the use of CB-SeC in a library reminder system. The other is a smart to do list manager project, decks, under development at Parallelism and Artificial Intelligence (PAI) Research Group at the University of Fribourg

[Lee and Helal, 2003] C. Lee and A. Helal. *Context Attributes: An Approach to Enable Context-awareness for Service Discovery*. Third IEEE/IPSJ Symposium on Applications and the Internet, Orlando, Florida, January 2003

[Maamar, et al, 2002] Z. Maamar, B. Benatallah, and Q. Z. Sheng. *Towards a Unified Composition Framework for E-M Services*. In the proceedings of the First AAMAS Workshop on Ubiquitous and Embedded wearable and mobile devices, Bologna, Italy 2002.

[Bhaskaran et al, 2002] Bhaskaran. R, Sharad A, and al. *The SAHARA Model for Service Composition across Multiple Providers*. In the proceedings of Pervasive Computing, Zurich, Switzerland, 1-14, August 2002.

[Vaughan, 2002] Vaughan-Nichols, S.J *Web Services, Beyond the Hype*. IEEE Computer 2002.

[Jennings and Wooldridge, 1998] N. R. Jennings and M. Wooldridge. *Applications of Agent Technology*. Editors, Agent Technology: Foundations, Applications, and Markets. Springer-Verlag, March 1998.

[O'Hare and O'Grady, 2003] G. M. P. O'Hare, M. J. O'Grady *Gulliver's Genie: A Multi-Agent System For Ubiquitous and Intelligent Content Delivery*. Computer Communications, Elsevier Science B.V, 2003.

[Kouadri et al, 2003] S. Kouadri.M and B. Hirsbrunner. *Towards a Context Based Service Composition Framework*. In the proceedings of the 1st International Conference in Web Services, ICWS'03, Nevada, Las Vegas June 2003.

[Schilit and Theimer, 1994] B. Schilit and M. Theimer. *Disseminating Active Map Information to Mobile Hosts*. IEEE Network, 8(5), September/October 1994.

[Ward, Jones and Hopper, 1997] A. Ward, A. Jones, and A. Hopper. *A New Location Technique for the Active Office*. IEEE Personal Communications, 4(5), October 1997.

[Dey et al, 2001] A. K. Dey, G. D. Abowd, and D. Salber. *A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications*. Human-Computer Interaction Journal, Special Issue on Context-Aware Computing, 16(1), 2001.

[Chen and Kotz, 2000] G. Chen, and D. Kotz. *A Survey of Context-Aware Mobile Computing*, Research. Dartmouth Computer Science, Technical Report, TR 2000-381, 2000.

[Khedr et al, 2002] M. Khedr, A. Karmouch, R. Liscano, and T. Gray, *Agent-Based Context-Aware Ad hoc Communication*, Proceedings. of the 4th International Workshop, MATA 2002, Barcelona, Spain, October 23-24, 2002.

[Lassila and Swick, 1999] O. Lassila and R. R. Swick. Resource Description Framework (RDF): Model and Syntax Specification. Recommendation. World Wide Web Consortium, Feb. 1999.

[Held et al, 2002] A. Held, S. Buchholz, A. Schill. Modeling of Context Information for Pervasive Computing Applications, Proceedings. of the 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI2002), Orlando, FL, Jul 2002.

[Courant et al, 2003] M. Courant, A. Tafat, B. Hirsbrunner. A generic Approach of Coordination for Ubiquitous Computing. Submitted to Smart Object Conference, Soc03, Grenoble, France, May 15-17, 2003.

[Chakraborty et al, 2002] D. Chakraborty, F. Perich, A. Joshi, T. Finin, Y. Yesha. A Reactive Service Composition Architecture for Pervasive Computing Environments. In proceedings of the 7th Personal Wireless Communications Conference (PWC 2002). Singapore. October. 2002.

[Jackson, 1995] M. Jackson. The world and the machine. In proceedings of the 17th International Conference of Software Engineering, (283-292), Seattle, Washington, USA April 24-28, 1995.

[Kouadri and Brézillon, 2003] G. Kouadri. M and P. Brézillon. A generic framework for context-based distributed authorizations. In Proceedings of the fourth International and Interdisciplinary Conference on Modeling and Using Context, Stanford, California (USA), June 23-25, 2003.