Investigating the Specifics of Contextual Elements Management: The CEManTIKA Approach

Vaninha Vieira[1,2], Patrícia Tedesco[1], Ana Carolina Salgado[1], and Patrick Brézillon[2]

[1] Center of Informatics, Federal University of Pernambuco, PO Box 7851, Recife, PE, Brasil
{vvs,acs,pcart}@cin.ufpe.br
[2] LIP6, University of Paris 6, 104 Avenue du Président Kennedy, 75016 Paris, France
{vieira,brezil}@poleia.lip6.fr

Abstract. In times where users need to process an ever increasing amount of information to perform more complex tasks in less time, the introduction of context in computer systems is becoming a necessity. However, building a context-sensitive system entails more work in comparison to traditional systems development: in the former, one must care for context-related tasks, such as the acquisition, processing, storage, manipulation and presentation of contextual elements. Context management proposes the separation of context related tasks from the application's business features. This paper presents a study on context management and discusses our proposal for a context manager, called CEManTIKA (*Contextual Elements Management Through Incremental Knowledge Acquisition*). A scenario is presented to illustrate its applicability.

Keywords: Context, Context Management, Context-Sensitive System.

1 Introduction

Users of computer systems often have problems to carry out their tasks effectively, since they have to process an ever increasing amount of information to perform more complex tasks in less time. Computing systems, in general, are not proactive, and usually act the same way and provide the same answers without considering the differences between their users. Hence, developers are seeking ways to build applications that are more adaptive, flexible and easy to use. The goal is to provide services that transparently ease the interface between humans and machines.

Context is what underlies the ability to differentiate one situation from another and to characterize entities and events. Differently from human to human interaction, where context is a well known and understood concept, to model and manipulate context in human to computer interactions is not a trivial task. Besides, context management generally is not the main feature but an optional, secondary functionality in a system. Also, most context-sensitive systems do not take into account requirements such as modularity, reusability or interoperability [1] and implement context manipulation tasks in a proprietary way, so as to fulfil the particular needs of each system.

B. Kokinov et al. (Eds.): CONTEXT 2007, LNAI 4635, pp. 493–506, 2007.
© Springer-Verlag Berlin Heidelberg 2007

Context management aims at providing solutions to separate context manipulation tasks from applications' business. This enables systems to reuse solutions and to share contextual elements. Generally, a context manager is responsible for context-related tasks such as: the acquisition, representation, processing, storage and dissemination of contextual elements.

In this paper we discuss the topic of context management and present our proposal for a context management system, named CEManTIKA (*Contextual Elements Management Through Incremental Knowledge Acquisition*). What differentiates our approach from others proposed in the literature (e.g. [2-6]) is that the existing ones restrict the contextual elements to those that can be perceived by sensors, such as location, identity, devices and activities. Since context is complex and dynamic, we believe that it is impractical for a system analyst to define *a priori* all contextual elements that must be managed by the system. Moreover, not all contextual elements that should be considered in the system can be perceived by sensors.

In this light, CEManTIKA proposes the incremental acquisition of contextual elements according to the usage of the context-sensitive system and addresses two main issues: (1) the definition and management of as much contextual elements as possible in the application domain; (2) the identification of the best way of using how to use these contextual elements to assist a specific situation distinguishing the set of relevant contextual elements.

The rest of the paper is organized as follows: Section 2 discusses the definition of context that we use throughout the paper; Section 3 introduces the topic of context management; Section 4 introduces the CEManTIKA system, discussing how it intervenes in context dynamics, presenting its architecture and exemplifying its usability through an example of use; Section 5 discusses some related works comparing them to our approach; and, Section 6 points out some final considerations and further work.

2 Our Working Definition of Context

In this work we use the context definition proposed by Brézillon and Pomerol [8]. According to their model, context is always related to a focus (e.g. a task or a step in a problem solving or decision making). At a given focus the context is the aggregation of three types of knowledge: *Contextual Knowledge* (CK), *External Knowledge* (EK) and *Proceduralized Context* (PC). CK is the part of the context that is relevant for the current focus while EK is the part that is not relevant. EK includes the knowledge unknown by the user or system while in the focus. The CK is the knowledge known by the user that is relevant to the focus at hand, and that could be activated to support the user in that focus. CK acts as a filter that defines, at a given time, what knowledge pieces must be taken into account (explicit knowledge), separating them from those that are not necessary or that are already shared (implicit knowledge). The PC is the dynamic part of the context; the one that will be effectively used in the focus. The PC set is composed by a subset of the CK that is assembled, organized and instantiated to address the current focus and effectively support the task at hand.

From a conceptual point of view, it is important to consider the EK, CK and PC whenever thinking about context, since the transformation of the EK into the CK is

related to the context dynamics. Not all contextual knowledge is instantly known in a focus, and to support the task it may be necessary to look for information that helps establish the context (acquired from the EK set).

Thinking in terms of implementation, we use the term *contextual element* (CE) to refer to pieces of data, information or knowledge that can be used to define the context. This separation is necessary because contextual knowledge, in fact, comprises what is in the user's mind and thus is too abstract. In order to treat context computationally, it is important to make this distinction between contextual data, contextual information and contextual knowledge.

Contextual data is the basic, atomic part of the context that can be acquired directly through virtual or physical sensors, such as location coordinates, people's identification or weather temperature. *Contextual information* is the CE that can be derived from several contextual data through association. For example, the contextual data set {[location=Paris], [month=01], [temperature=18°C]} implies the contextual information [weather=hot]. If we change the location to Recife (Brazil) the equivalent contextual information will be [weather=cold]. So, we may have the same set of data with different interpretations: While the information is something that once inferred can be easily instantiated and shared between human and software agents, the *contextual knowledge* is personal and is inside people's head as mental schemas that help them to interpret external events.

We identify and manipulate CEs in terms of CE Sets. A *CE Set* comprises one CE and its instances. Furthermore, a CE may have one or more instances, which are themselves CEs. For example, consider a CE [MissionOfficialReasons] that has as instances the set {[Presentation], [Experimentation], [Meeting]}, meaning that the official reasons of an academic mission may be for a presentation, an experimentation *in loco* or a meeting. In its turn, the instance [Presentation] is itself a CE and has as instances the subset {[Seminar], [ConferencePaper]}.

Another term we use in our work is Instantiated CE (*ICE*). An ICE means the chosen instance for a CE in a given focus by a specific user. For example, a user Luce in her mission identified that the reason for her mission was to present a paper in a conference. So the ICE for Luce related to the CE [MissionOfficialReasons] is the path {[Presentation]/[ConferencePaper]}. An ICE may also include a value for a CE (e.g. {[MissionLocation]="Paris"}. An *ICE Set* is used to define the PC in a focus, meaning the set of all relevant ICE in that focus.

Because a CE can itself have a context, we meet McCarthy's observations [9]: (1) a context is always relative to another context, (2) Contexts have an infinite dimension; (3) Contexts cannot be described completely; and (4) When several contexts occur in a discussion, there is a common context above all of them to which all terms and predicates can be lifted.

3 Context Management

Context-sensitive systems are those that understand and use contextual elements to provide relevant services and/or information to the users or to other applications during the execution of some task. The building of a context-sensitive system comprises tasks such as: to specify the CEs needed in the application domain; to build

components to acquire and instantiate these CEs; to create aggregation and reasoning modules that enable it to process the instantiated CEs according to the application needs; and to effectively use the identified set of instantiated CEs in a focus to provide for the application adaptability.

Context management involves the definition of models and systems to assist the acquisition, manipulation and maintenance of a shared repository of CEs, thus enabling the usage of these elements by different context-sensitive systems. The main idea is to reduce the complexity of building context-sensitive systems, by transferring tasks related to CE manipulation to an intermediate layer. In this light, the task of managing context includes the definition of: (1) a representation model to describe and share CE sets; (2) an infrastructure to detect, update and query CE sets; (3) mechanisms to process, reason about, and infer new CE sets from existing ones; and (4) mechanisms to identify the ICE in a focus.

An overview of the context management process and its main functionalities is illustrated in Fig. 1. The first step is to *acquire* the CEs associated to a situation. Computer systems may use virtual and physical sensors, user interfaces (e.g. forms), persistent databases, and so on, to acquire these elements. After that, the system must use knowledge bases, and inference engines to *process* the acquired CEs through reasoning and associations. The interpreted context is used to infer information and to trigger services that must be provided and executed.

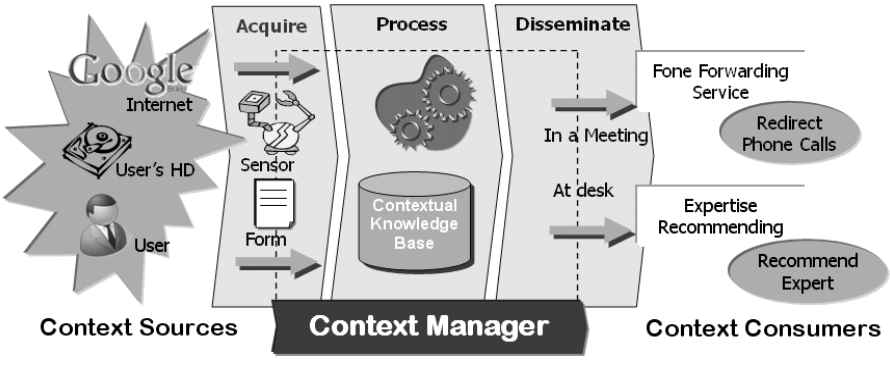

Fig. 1. Overview of a Generic Context Manager Main Functionalities

For example, the context manager may acquire information about a given user's presence, activities scheduled in her/his calendar and location from context sources such as web search engines, message exchanging programs (e.g. MSN and Yahoo), the user's hard disk, or the user her/himself. The acquired elements are then processed and the manager may infer that "the user is available at her/his desk". With this inferred CE, a Phone Forwarding Service can redirect phone calls to the user's desk. If the manager infers that the user is in a meeting, than the phone forwarding service may redirect the phone calls to the user's mobile phone or to her/his secretary. The example described before is about functionalities that may be provided by any

context-sensitive system. However, a context management system brings the added advantages described below:

- *Reusability*: the solution for each context management task can be done in a generic way and be reused by several applications;
- *Sharing*: Applications can share CEs acquired from different and heterogeneous context sources;
- *Context source independence*: Applications are developed independently from the underlying contextual source;
- *Ease of use*: Application developers can focus on their business model and leave details of context management to the manager implementation.

In order to be effective, a context manager must take into account aspects such as: separation of the context model and the application domain model; maintenance of a sharable context model that enables communication between different components or systems; provision of descriptions and interfaces for the manager internal components and their formats to allow communication and interoperability among the manager's components and context consumers.

4 Managing Contextual Elements with CEManTIKA

This section presents our proposal for a context manager, named CEManTIKA. We discuss an overview of the system, its relation with context dynamics, and its architecture. A more complete description of the system can be found in the project site [10].

4.1 CEManTIKA and Context Dynamics

Since context is what enables the characterization of entities (e.g. people, devices, actions, events, software components and so on) and entities exist within a knowledge domain, context is also strongly influenced by the domain it is applied in. This means that when introducing and managing CEs for an application domain one must first identify the CE Sets that characterize the entities in that domain, and construct the Contextual Elements Base (CEB) for the domain. Different domains necessarily entail building different CEBs since the CE could have different meanings in different domains.

Context is a dynamic construction that evolves with the focus. As the focus changes, the set of CEs that must be considered changes accordingly. Context dynamics is represented by the transformation of External Knowledge into Contextual Knowledge and then into Proceduralized Context influenced by the focus as stated in Brézillon and Pomerol's model [8] and illustrated in Fig. 2. CEManTIKA manages the different focus in the domain and, for a given focus, it identifies which CE Sets must be considered and instantiated to support the task at hand (the ICE Set). A Proceduralized Context Base (PCB) maintains historical cases of the ICE Set built and their respective focus. The historical ICE Sets stored in the PCB aid the identification of the relevant CEs in other focus.

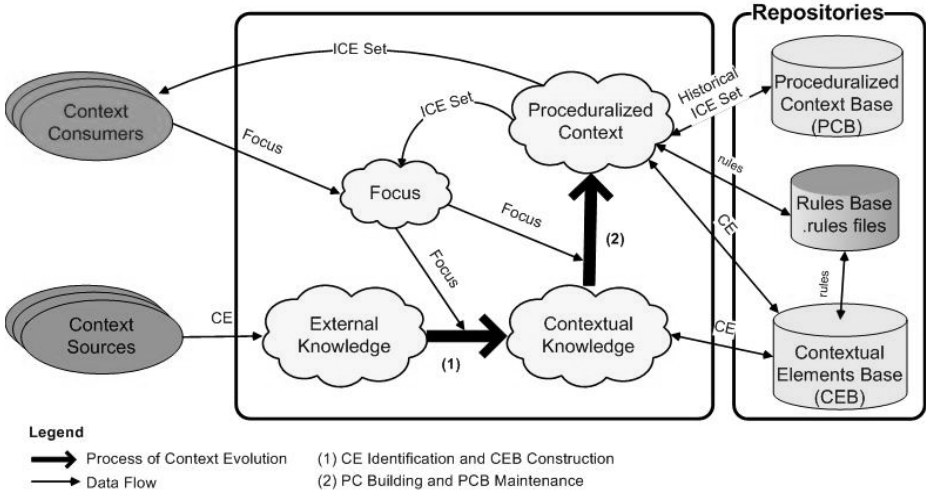

Fig. 2. The Role of CEManTIKA in Context Dynamics

The context management in CEManTIKA comprises two main processes (as illustrated in Fig. 2): (1) the CE Identification and CEB Construction; and (2) the PC Building and PCB Maintenance. These processes are detailed in the next sections.

4.1.1 CE Identification and CEB Construction

This process comprises the identification of the CEs in a domain and the acquisition of CEs from different context sources by CEManTIKA. Since context is a very subtle concept and has an infinite dimension, it is impossible to think about context management in a totally generic way without defining and limiting the scope of what will be characterized. So, it is important to identify and delimit the domain and the entities that the contextual elements are about.

To be effective, a context manager must know the domain entities very well. Thus, a well designed and filled in CEB is a key factor. It is necessary to look increasingly deeper and specifically in the domain , so that the manager can identify how a change in the context affects the state of the entities and consequently the system's actions and events.

To understand this requirement we can think about a GPS system, a very successful example of context-sensitive system, which aids drivers to identify the best itinerary from one place to another. As dynamic contextual element, GPS consider the user's location at a given moment. According to that location the system shows the path that the user must follow to arrive at the destination; as the user moves on, the GPS updates this path and indicates the next steps that the user must take until finally arriving at the destination. To provide the appropriate itineraries the system counts with as much knowledge as possible related to the region it will be used in. For instance, a base with information about France will be really useless in Brazil.

The difficulty of identifying the CE Sets in a domain is that the interpretation of the contextual elements changes widely, varying according to different systems' users. Hence, it is very difficult and not very reliable for a system designer to describe

a priori all contextual elements related to a domain/task based exclusively on her/his own experience. We believe that the CE Sets must be identified and defined incrementally during the system usage with the users' participation. As stated by Brézillon in [7], the incremental acquisition of CEs may occur either by: the acquisition of new knowledge pieces; the learning of a new knowledge structure while building the PC; or learning by structuring the contextual knowledge through the user's feedback after using the proceduralized context. Thus, the manager will be able to learn and accumulate several views from different people acting as a kind of memory support system.

4.1.2 PC Building and PCB Maintenance

The PC building process is related to the identification of the CE and ICE Sets that must be considered to support the current focus and to the maintain the PCB. Here, CEManTIKA acts in four points:

(i) Given a focus, CEManTIKA identifies and extracts from the CEB the CE sets that should be considered;
(ii) After that, using rules defined in a rule base, CEManTIKA selects a CE subset that is instantiated and that will be used to build the PC in the focus (the ICE set) from the identified CE sets . To build the PC, the manager also considers the historical ICE sets maintained in the PCB;
(iii) The manager gives the user the opportunity to identify if the chosen CEs were really useful, allowing the user to validate the PC built in the ICE Set.;
(iv) The last point is the manager's ability to learn from the user's feedback, which entails the following activities: building a new ICE Set including and instantiating additional CEs necessary; or changing the current focus and rebuilding the ICE Set according to this new focus.

The ICE Set thus built affects the focus and may demand the instantiation of other CEs or the inclusion of new CEs (acquired from context sources) into the system. For example, consider a scenario of a system that support users in planning travels, where the user's focus is *booking a hotel*, and the current ICE Set is {[hotelType]=[comfortable]; [hotelLocation]=[near the conference]}. According to this ICE Set the system proposes a list of hotels that fits the user's current context. Later, when the user's focus is *verifying mission costs*, s/he observes that the nearest hotels are also the most expensive and so s/he cannot afford to pay for them thus, another CE {[maxPrice]<[availableHotelResource]} is inserted into the ICE Set. Thus, when the user is again in the focus *booking a hotel* this new ICE Set will be used; allowing the system to build a new list of hotels considering also this new CE.

4.2 CEManTIKA Architecture

An overview of the CEManTIKA architecture is presented in Fig. 3. CEManTIKA components are located in two different hosts: (1) *the context-aware system host*, where the context-sensitive system that uses the manager is running; (2) *the CEManTIKA server host* that maintains the core components and the repositories (CEB and PCB). The context sources and consumers (components in light gray in Fig. 3) are the interaction points between the context-sensitive system and CEManTIKA.

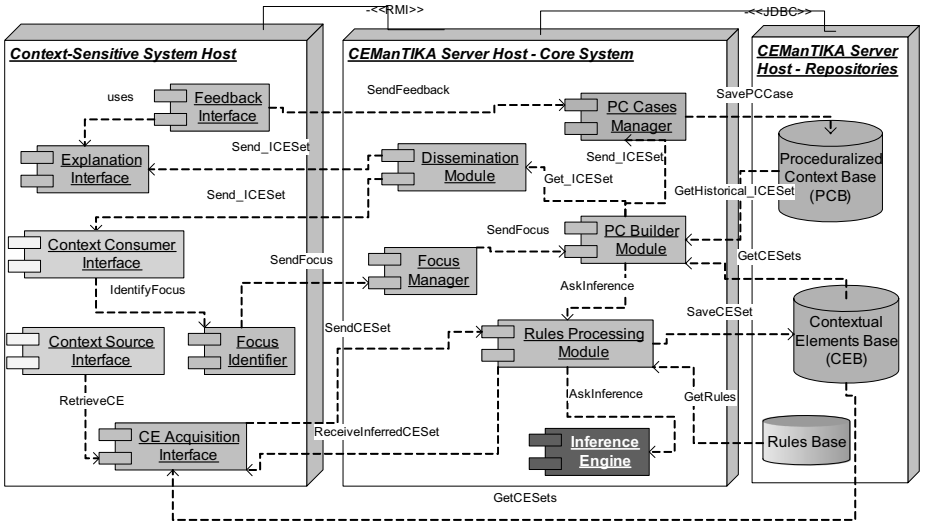

Fig. 3. Overview of CEManTIKA Architecture (using UML notation)

The process described in Section 4.1.1 is performed by an instantiation of the component *CE Acquisition Interface*, attached to a context source. Each context source is associated to one or more acquisition components. The acquired CE Sets are sent to the *Rule Processing Module* that uses the pre-defined rules and an inference engine to process the CE sets, verifying inconsistencies, and inferring new CE sets that are sent back to the acquisition module.

The process discussed in Section 4.1.2 in the items (i) and (ii) occurs as follows: inside the client host, the component *Focus Identifier* discovers what the user's current focus is and sends this information to the *Focus Manager.* The Focus Manager then informs the current focus to the module *PC Builder,* which triggers the generation of a new ICE Set. To do this the *PC Builder* identifies what are the CE Sets associated with the focus in the CEB and retrieves their instantiated values. Also, the PC Builder looks in the PCB for historical ICE Sets built before for the same focus. With these two inputs the PC Builder decides which CEs will be considered and builds the final ICE Set. After that, it sends the ICE Set to the *Dissemination Module* that will redistribute it to the *Context Consumer* and the *Explanation Interface*. The Explanation Interface provides the user with an explanation about how the ICE set was built, including what rules were activated or what decisions were made to restrict the set of CEs that was finally considered. Through the *Feedback Interface* the user (or a software agent) can indicate to CEManTIKA how useful was the ICE Set for her/him during his task development and how much the ICE Set has really expressed her/his current context. This feedback is sent to the *PC Cases Manager* that stores the new case in the PCB for later usage.

The current version of CEManTIKA is implemented in Java (the core components) and PhP (the users' web interfaces). As communication protocols we are using: RMI (*Remote Method Invocation*) to enable the communication between the client and

CEManTIKA Server, and JDBC (*Java Database Connectivity*) as interface with the repositories host. The repositories are currently described using the relational model in MySQL databases. And, as reasoning technique we are using first order logic through production rules that are processed by the inference engine JEOPS (Java Embedded Object Production System [11].

4.3 Example of CE Manipulation

To illustrate the functioning of CEManTIKA, let us consider the domain of academic missions. Researchers frequently have to plan their travels to accomplish missions in other organizations, generally located in different cities. These missions may have different purposes, such as to present a paper in a conference, to perform research experiments or to participate in a meeting. The mission planning demands that the researcher takes care of several tasks; some of which are illustrated in Fig. 4: (F1) provide the general description of the mission; (F2) book transport; (F3) book accommodation; and (F4) execute the procedures to effectuate the payment. Each task has a specific objective and represents a different focus for the user.

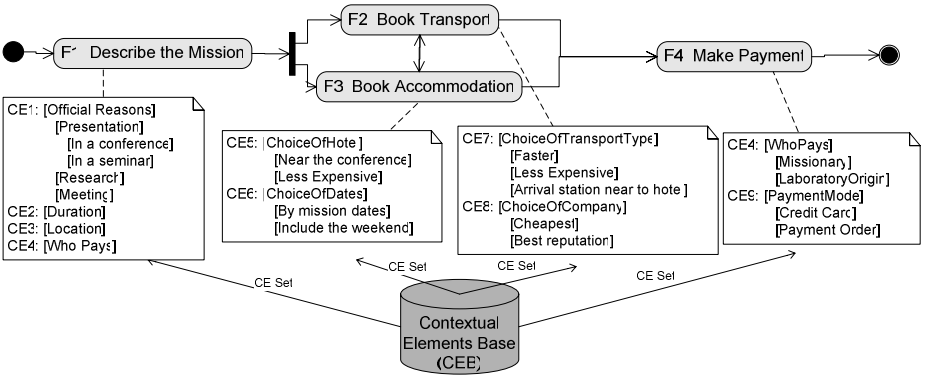

Fig. 4. Illustration of some foci and related CE sets in a mission planning

Each focus activates a different set of CEs. In the example, the focus F1 has the following CE Set {[Official Reasons]; [Duration]; [Location]; [Who pays?]}. Since, a CE may instantiate others CE Sets, the CE [Official Reasons] instantiates the CE Set {[Presentation]; [Research]; [Meeting]}, and so on.

In this domain we have specified a context-sensitive system called Form Filling Support Agent (FFSA). FFSA is an assistant that aids missionaries in their mission planning through a dynamic form that follows the user through the different phases (i.e. focus) related to the mission. FFSA interact with CEManTIKA to obtain the corresponding CE Sets given a new focus. Through the instantiated CE sets FFSA can, for example, emphasize the fields in the form the user must fill or suggest related information that could interest the user and support her/his decisions.

As an example we can consider a missionary named Luce, which is as a professor at the London United University. Luce must prepare a new mission to Recife (Brazil)

to present a paper in a conference and she uses the FFSA to assist her in this mission planning. The mission form is composed initially with the CE Sets stored in the CEB, associated to each focus (as illustrated in Fig. 4), and each focus activates a different CE Set. During the form filling Luce can instantiate the existing CE Sets with information about her mission. Fig. 5 illustrates the ICE Set related to Luce's mission in the focus F1: {[Official Reasons]/[Presentation]/[In a conference]; [Duration]=[less than a week]; [Location]=[Brazil/Recife]; [Who Pays]=[London United University]}.

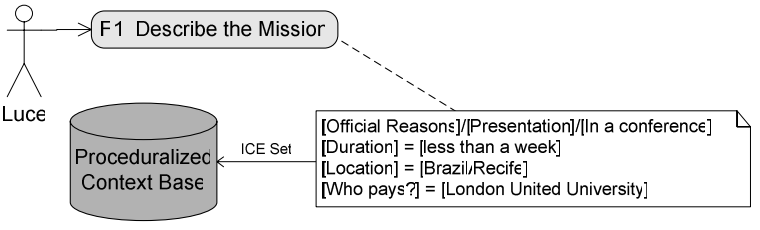

Fig. 5. Example of an ICE set related to a user in a focus

While Luce fills in the mission form, the FFSA uses the knowledge contained in the ICE Set to find additional information in Luce's hard disk (e.g. the hotel she stayed in her last trip to Recife) or over the internet (e.g. available companies and current tickets' prices from London to Recife, hotels near the conference or money exchange information). So, the analysis of the ICE Set built in the focus enables the FFSA to determine the actions it must execute. Additionally, the ICE Set can provide new CEs that will support the building of the ICE Set in another focus. For example, in the focus F2 [Book Accommodation] the list of suggested options for accommodation will change according to what the user filled in the focus F1 (the mission description): if the [Duration]=[less than a week] then an accommodation in a hotel will be just fine, but if [Duration]=[several months] then a hotel will really not be a good option, instead an apartment to rent will be better; and if [Duration]=[several years] the user could also consider to buy a property (apartment or house).

This is a small example of how a context-sensitive system may benefit from the management of CEs with CEManTIKA, through the manipulation of the user's current focus and the corresponding ICE Sets. Besides, integrating the context manipulation with the system functionality seems to be a good approach, since the manager may "learn" the contextual elements while the user executes her/his tasks.

5 Related Works

The proposals for context management that appear in the literature are associated to toolkits [2], frameworks [3], middlewares [6], engines [4] and specifications [5]. We selected one work representative of each category in different application areas to describe and compare them with our proposal.

5.1 Existing Approaches for Context Management

SOPHIE [4] is a reactive and integrated information environment that tracks the constant changes in an environment to adapt to them, for example, through the dissemination of the correct information to different receivers. As context model it uses an extension of ORM (*Object-Role Modeling*) [12] to define context concepts in high levels of abstraction. SOPHIE is integrated to a context engine that has four main layers: *context sensing*, to acquire contextual information; *context augmentation*, to store contextual information associating them to the related subject; *contextual adaptation*, to adapt the system behavior according to changes in the current context; and *contextual resource discovery*, to discover relevant context-dependent information resources. Contextual information is acquired from two sources: the application level (information stored in databases); and the environment level (dynamic information about the real, physical world).

SOCAM is a middleware for rapid prototyping of context-aware services in intelligent environments [6]. As a context model it uses an ontology named CONON (*Context Ontology*), which represents concepts related to locations, users, activities and computational entities. To assist context management, SOCAM provides support for context acquisition, sharing, reasoning, storage and dissemination. A knowledge base is used to store the acquired and inferred contextual knowledge. A service-locating service provides ways for context providers to publish the context information they can provide.

CXMS is a framework that offers a tool set to ease the development of context-sensitive systems [3]. They consider the context to be composed by elements such as identity, location, time and environment, represented by key-value pairs. A *context toolkit* implements the context management through four tasks: *context acquisition*, by the sensor layer; *context modeling*, through the semantic layer that provides context interpretation, semantic enrichment and context evolution management; *application adaptation behavior definition*, through the control layer, which decides the actions that must be executed in particular conditions; and *adequate information presentation*, through the actuation layer, which maps the decisions made by the control layer to actions.

CMA [5] is a standard specification for a Context Management Architecture applied to the domain of Clinical Applications. It is intended to manipulate patient contextual data, as part of the CCOW (*Clinical Context Object Workgroup)* [5]. CMA defines the interfaces and privacy policies between clinical applications known as context participants and a context manager. Context Participants query the context manager when they need to determine the current context and when they wish to update the patient's context. A CCOW-based context manager should be implemented according to the specifications in order to support different applications providing a unique sign-on portal, enabling patient's context to be propagated and interchanged between these applications. Examples of commercially available products implementing the CCOW specification are the Sentillion Vergence, Carefx Fusion and Orion Health's Concerto Context Management Suite [5].

5.2 Comparison with Our Proposal

In general, context managers take care of tasks such as: context acquisition, representation, processing, storage and dissemination. The main difference between our proposal and the approaches found in the literature is the way of reasoning about context and, consequently, the way of processing and managing it. Other approaches restrict the type of CE managed too much, limiting it, in general, to those that can be automatically acquired from physical and logical sensors (e.g. location, identity, devices and activities). Even activities are also limited to a small set of predefined options. Moreover, such managers do not provide much flexibility to change the types of CEs initially considered and managed. Another difference is that other managers do not consider the dynamics of context and generally reason in static terms.

The problem with managing context is that context is a very complex and dynamic concept, comprising much more knowledge than what sensors can perceive. We propose a context manager that can increase incrementally the types of managed CE according to the usage of the context-sensitive system, thus enabling context evolution. The manager defines the procedures and infrastructure to manipulate the CE independently from the domain or from the focus at hand. In this light, CEManTIKA is not limited to a static and pre-defined set of CE and can be reused in different domains by different applications.

6 Conclusions and Further Work

Context is becoming a necessity in computer systems. However, building a context-sensitive system entails high development cost because several context-related tasks (e.g. context identification, representation, acquisition, processing, storage, and usage) must be addressed by systems developers. Context management systems propose the separation of context manipulation tasks from the applications' business features, enabling modularity and reusability, to facilitate the building of context-sensitive systems.

This paper presented a study about context management and our proposal for a context management system, named CEManTIKA. CEManTIKA is centered around two main features: (1) to provide a domain-independent context manager that considers the dynamic nature of context, enabling the flexible and incremental building of a contextual elements base; (2) to promote the use of the current focus and case-based techniques to identify and instantiate the relevant contextual elements to support the task at hand (the Proceduralized Context).

What differentiates CEManTIKA from other approaches is that the latter restrict the contextual elements to those perceived by sensors, such as location, person identity, devices and activities. Since context is a complex concept we believe it is a better approach to enable the incremental construction of the contextual elements base according to the usage of the context-sensitive system. Also, we propose a generic, domain-independent manager that is based on a well known and accepted conceptual view of context, which guarantees that changes in the domain will not influence how the contextual elements are manipulated.

Although we understand the need to consider the interaction between the context manager and the user, this paper focused in describing the general ideas behind the manager, discussing in detail how the manager intends to support the dynamics of context evolution through the building of CEs and ICE Sets. Our studies assure us that a context manager leads to a flexible organization of the context in a changing focus, more powerful than the current fixed and rigid structures of database management systems.

Currently, we are working on the implementation of the example of use described in this paper, which includes the construction of a CEB in the academic mission domain and the prototype of the Form Filling Support Agent (FFSA). Also, we are implementing an External Information Retrieval Agent (EIRA), which makes contextualized searches in the internet for complementary information related to the mission that is being planned. The mission form will be a web page accessible by different researchers. The EIRA will provide search services in the web for transports and accommodation that match the mission values provided by the user. Changes in the focus and in mission context implies in different criteria of search by the EIRA.

We are also working on the validation of the generality of our approach by the instantiation of CEManTIKA components to the domain of expertise recommending with ICARE, a context-sensitive expert recommending system [13]. For the CE acquisition features, we are experimenting the usage of different acquisition agents attached to existing and popular working tools such as Microsoft Office (e.g. Word, Excel, PowerPoint and Outlook) and instant messenger systems (e.g. MSN). In the near future we plan to rewrite the CEB using ontologies instead of the current relational model. This is important because ontologies enable: the sharing and reuse of CE definitions between different domains; the explicit representation of the reasoning specification with its respective CE; and the usage of different existing inference engines to reason over the defined CE Sets [14].

Acknowledgments. First author wants to thank CNPq and CAPES for their financial support, and the UFBA for its support.

References

1. Riva, O.: A Context Infrastructure for the Support of Mobile Context-Aware Services (2005), Accessed in 03/2007, http://www.cs.helsinki.fi/u/kraatika/Courses/f4fMC/WS1/Riva.pdf
2. Dey, A.K., Salber, D., Abowd, G.D: A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Human Computer Interaction Journal 16, 97–166 (2001) Special Issue on Context-Aware Computing
3. Zimmermann, A., Lorenz, A., Specht, M.: Applications of a Context-Management System. In: Dey, A.K., Kokinov, B., Leake, D.B., Turner, R. (eds.) CONTEXT 2005. LNCS (LNAI), vol. 3554, pp. 556–569. Springer, Heidelberg (2005)
4. Belotti, R.: Sophie - Context Modelling and Control, Diploma thesis, Swiss Federal Institute of Technology Zurich (2004).
5. Seliger, R., Sentillion.: HL7 Context Management CCOW Standard: Technology and Subject-Independent Component Architecture (2003), Accessed in 03/2007, http://www.hl7.org.au/CCOW.htm

6. Gu, T., Pung, H.K., Zhang, D.Q.: A Service-Oriented Middleware for Building Context-Aware Services. Elsevier Journal of Network and Computer Applications (JNCA) 28(1), 1–18 (2005)
7. Brézillon, P.: Task Realization Models in Contextual Graphs. In: Dey, A.K., Kokinov, B., Leake, D.B., Turner, R. (eds.) CONTEXT 2005. LNCS (LNAI), vol. 3554, pp. 55–68. Springer, Heidelberg (2005)
8. Brézillon, P., Pomerol, J.-C.: Contextual Knowledge Sharing and Cooperation in Intelligent Assistant Systems. Le Travail Humain, PUF, Paris 62(3), 223–246 (1999)
9. McCarthy, J.: Notes on Formalizing Contexts. In: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, San Mateo, California, pp. 555–560 (1993)
10. Vieira, V.: The CEManTIKA Project Homepage (2007), Accessed in 03/2007, http://www.cin.ufpe.br/~vvs/cemantika/
11. Figueira Filho, C.: JEOPS - The Java Embedded Object Production System (1999), Accessed in 04/2006, http://www.cin.ufpe.br/~jeops/
12. Henricksen, K., Indulska, J., Rakotonirainy, A.: Generating Context Management Infrastructure from High-Level Context Models. In: Chen, M.-S., Chrysanthis, P.K., Sloman, M., Zaslavsky, A. (eds.) MDM 2003. LNCS, vol. 2574, pp. 1–6. Springer, Heidelberg (2003)
13. Petry, H., Vieira, V., Tedesco, P., Salgado, A.C.: Um Sistema de Recomendação de Especialistas Sensível ao Contexto para Apoio à Colaboração Informal. In: Simpósio Brasileiro de Sistemas Colaborativos, Natal, RN, pp. 38–47 (2006)
14. Vieira, V., Tedesco, P., Salgado, A.C.: Towards an Ontology for Context Representation in Groupware. In: Fukś, H., Lukosch, S., Salgado, A.C. (eds.) CRIWG 2005. LNCS, vol. 3706, pp. 367–375. Springer, Heidelberg (2005)