Context-Aware Information Services for Health Care

Jens H. Jahnke, Yury Bychkov, David Dahlem, Luay Kawasme

Department of Computer Science
University of Victoria
Victoria, BC, Canada
[jens|ybychkov|ddahlem|lkawasme]@netlab.uvic.ca

Agility is a distinctive aspect of the health care profession. Clinicians operate in a fast changing environment with needs to access different pieces of information in different situations. On the surface it would appear that clinicians would benefit tremendously from the use of modern mobile technology. In reality, however, the health care sector has been hesitant to embrace electronic clinical information systems in general and mobile technologies in particular. To improve the adoption of mobile health information, we introduce an ontology-based Context Management System (CxMS) that allows the user to define contexts using terms from the medical field. In a highly sophisticated and complex field like health care, a generic or 'one-size-fits-all' context aware approach does not meet specialized requirements. The CxMS is extensible to allow the context aware system to develop and evolve in domain-specific ways.

1. Introduction

Health care professionals are generally reluctant adopters of modern health information systems. Whereas the health care industry is usually very quick to exploit the latest cutting edge medical technology, it is not unusual for supermarkets to have more sophisticated information services than hospitals [17]. In the United States, fewer than 5% of all physicians utilise electronic record systems in their practice [15], and about 10% of hospitals have made computerised physician order entry systems available [10]. Furthermore, according to [3], half of all computer-based information systems fail mainly due to user resistance and staff interference. The main complaint made against such systems is that users were asked to significantly alter traditional workflow patterns to accommodate the system, rather than the system accommodating the users. At the Cedars-Sinai Medical Center in Los Angeles, a recently installed electronic information system was disengaged because physicians complained that "rather than speeding up and improving patient care, it actually slowed down the process of filling their orders. [9]" This concurs with our experiences in implementing a point-of-care mobile information management system in a local health facility: end-users are slowed by inefficient modes of data input and lack of screen space on mobile devices.

A partial solution to the problem described above can be achieved through *context awareness*. For agile users like physicians, nurses, a great deal can be inferred about

their current task by where they are, the time of day, and with whom they are in close proximity. In our sample scenario, an Ophthalmologist may see a patient in four different situations: an initial assessment or consultation, a pre-operative visit, a surgical procedure, and a post-operative assessment. The information required for each situation is different as is the type of information that must be captured:

Visit Type	Location	Data Entry Requirements	Data Displayed
Initial Assessment	Physician's private office	Slit lamp report Corrected vision report Assessment report	Referring physician's diagnosis Patient history
Pre-Operative	Hospital	None	Recent blood work Current medications
Operative	Hospital	Operative report	X-Rays
Post-Operative Assessment	Physician's private office	Slit Lamp exam Corrected vision exam Post-Operative assessment report	Operative Report Referring physician's diagnosis Patient history

The usability of the system can be greatly enhanced by utilising contextual factors to display only the needed information and to provide optimised data entry screens, in accordance with the task of the user.

In this position paper, we discuss preliminary results of our research project on developing context-aware information services for health care applications. Despite the fact that our primary application domain is health care, we expect that many concepts discussed here can be transferred to other application domains.

The rest of this paper is structured as follows. We begin with a short review of related work. In Section 3 we introduce the conceptual architecture of our context-aware information system. In Section 4 we present a formal model (context ontology) for representing domain-dependent and domain-independent contextual information. After a brief discussion on how we propose to represent and store contextual facts, we conclude with a discussion about how we formalise context patterns in order to recognise canonical contextual situations.

2. Background and Related Work

While the field of context-aware computing is relatively young, it has been actively developing for the last decade. The term *"context-aware"* was coined by Schilit and

Theimer in [23], where the context was defined as "*location, identities of nearby people and objects, and changes to those objects"*. This definition has changed over the years (e.g. [5]) until Dey had redefined it in [13] as *"any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"*. Dey's notion of context fits very well with our ontology-based approach.

During this time a number of powerful context-aware systems such as Salber's, Dey's and Abowd's Context Toolkit [22] and Hewlett Packard's Cooltown [19] have been developed. Unfortunately, according to Chen et al. [7], most of these systems shared a weakness in *supporting knowledge sharing and context reasoning* because of their lack of ontologies. However, this weakness stared to be addressed by projects like Context Broker Architecture (CoBrA) [8] and GAIA [21]. Our COWSPOTS project continues this trend by using a partially domain-specific (for the healthcare domain) ontology.

Existing context-aware projects use a variety of methods for reasoning about context. Some (e.g. Context Toolkit [22]) just enable application to access the contextual information, but do not provide any other support. Castro at al. [6] used Bayesian networks to infer context data from sensor data. Ranganathan and Campbell [21] use both Bayesian and rule-based inference engines and specify behaviour of contexts using first order logic.

The range of applications for context-aware projects is rather wide. GUIDE [11] and Cyberguide [2] focus on providing the electronic equivalent of a tour guide. Context Toolkit [22] and GAIA [21] offer the frameworks and infrastructure for developing other context-aware applications, while iCAP [26] is used for the interactive prototyping on such applications. Closer to our application domain are the works of Bardram [4] and Davis et al. [12], that focus on the use of context-aware applications in healthcare.

A significant amount of research in this area has also been done on the hardware side of the context-aware computing. This research ranges from the use of sensors [24] to capture the context information and augmenting everyday objects in order to make them context sensitive [16] to wearable computers [1] and augmented reality [20]. However, while very interesting, this work is out of scope of our paper.

3. Context Management System

The purpose of this section is to introduce the overall architecture for our Context Management System (CxMS). Figure 1 shows the architecture for the CxMS. We use Microsoft's SharePoint Portal Server 2003 as a platform for our implementation. Analogously to similar products from IBM and Sun Microsystems, SharePoint supports the notion of *Web parts*, i.e., small sub-components of visual web pages that can be viewed with normal thin-client web browsers. In a context-aware, mobile application scenario as described earlier, users want to interact with different information services depending on the current situational parameters. By choosing SharePoint as the platform for our CxMS, we are provided with a predefined

framework for implementing those information services in terms of Web parts and composing context-aware Web pages from them. Figure 2 shows an example screenshot of the SharePoint-based context-aware information service. This particular view consists of four Web parts, which an ophthalmologist would like to see in a post-operative check-up context with a patient: basic patient information (top-left), Cataract Assessment Report (bottom-left), Operative Report (right-top), and Post-Op Report (right-bottom).

The left part of the browser window in Figure 2 shows additional historical information about the data presented in the different Web parts. For example, it shows that the Cataract Assessment report was entered by a referring physician, whereas the Operative Report was entered by the anaesthesiologist.

The actual composition of context-aware web pages from Web parts is performed by an extension to SharePoint, called *Context View Manager*. At the core of the CxMS are a *Context Fact Base*, a *Context Pattern Base*, and a *Context Inference Engine*. The context fact base is graph-based database storing facts about situational parameters of user interactions with the information services provided by the system. Contextual facts can be delivered by a variety of different types of *context sensors*, including but not limited to location sensors.

The context pattern base stores rules that associate certain Web parts with formal definitions of predefined usage contexts. The context inference engine infers actual usage contexts based on predefined context patterns and current data in the context fact base.

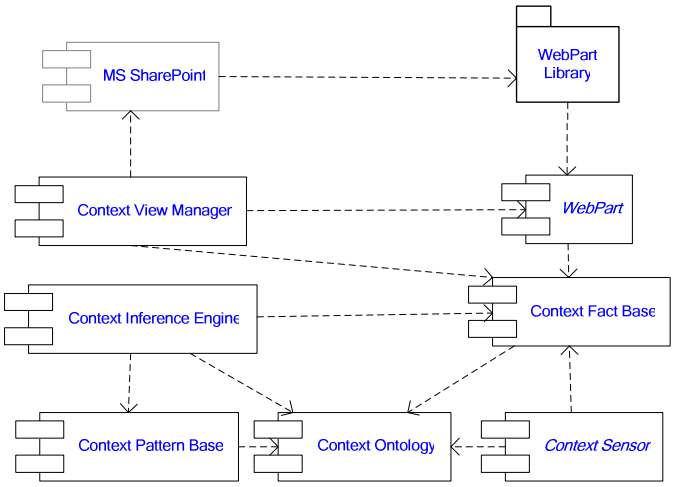

Figure 1. Architecture of Context Management System

All core components of the described architecture require a common theory on how to formally represent context. This theory is called *context ontology* and is discussed in the next section.

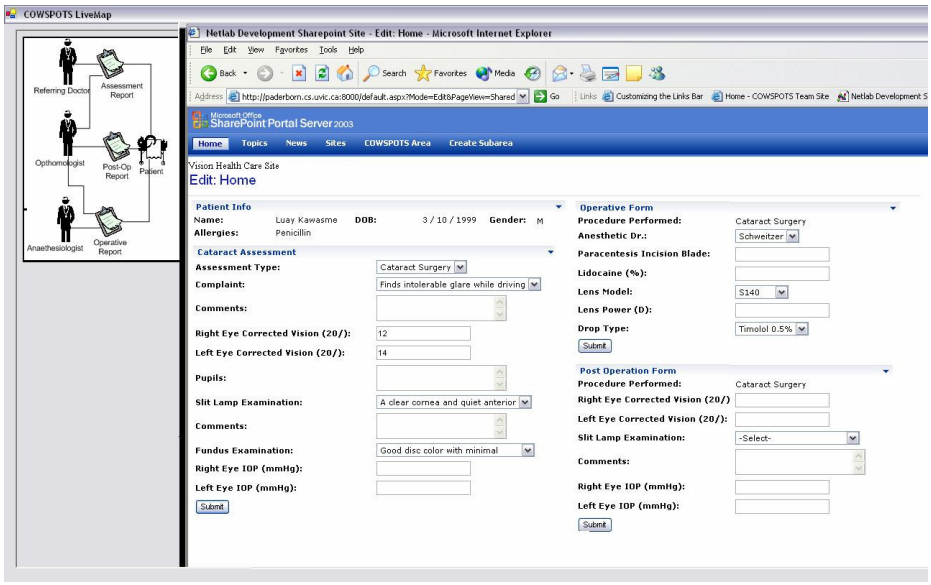

Figure 2. Context-Aware Information Services - Example

4. Context Ontology

The application domain for our context-aware information technology is health care. Therefore, it has to be targeted towards the concepts relevant in this domain. On the other hand, there are also domain-independent concepts in the context ontology, e.g., concepts for time and space. For the domain-dependent part of our context ontology, we have adopted existing domain ontologies in health care information management. The two most important domain ontologies we have considered are the HL7 Reference Information Model (RIM) and the HL7 Clinical Document Architecture (CDA)[1]. The purpose of the RIM is to describe and formalise all major concepts involved in health care, such as organisations, individuals, devices, places, but also acts and documents. Based on the RIM, the CDA defines how electronic health care information should be represented in structured documents. CDA documents can consist of several *sections*, which themselves consist of several information *entries*. For example, a CDA "patient chart" could contain several sections including a section on "blood work", which itself contains several entries such as "blood type" or "haemoglobin".

For the purpose of our research, we define the problem of context-aware delivery of health information as the problem of representing the right CDA documents for the current situational context. Consequently, our context ontology has to relate concepts of the CDA ontology with domain concepts in the RIM. Figure 3 shows a simplified version of this joint context ontology in UML-like notation. The domain-dependent

[1] www.hl7.org

part depicted at the top shows documents as specialisations of the *information* concept and consisting of sections and entries. The RIM-based model in which this information can be placed has concepts for *actors*, *roles, persons* and *devices*. The concept of an actor is used as a generalisation for uniquely identifiable *persons* (individuals), *roles* (abstract individuals), and *casts* (individuals in particular roles). The concept of a device represents a generalisation for medical devices (x-ray machines, CT scanners, etc.) as well as IT devices (computer workstations, tablet PCs, PDAs, Smart Phones etc.). *Delivery* relations associate IT devices with information services delivered to actors.

The bottom part of Figure 3 depicts the domain-independent part of the context ontology. This part should be re-usable for other application domains, since it covers only contextual concepts inherent in our physical world. It attaches a concept for a *position in space and time* to domain context elements. Our ontology only shows two of the many *spatio-temporal* relationships possible between such positions: A *syncro-colocation* relationship means that two positions are spatially co-located and synchronous in time, whereas a *preceding-colocation* would imply spatial co-location but temporal precedence.

Periods are defined by two points in time, which can be represented by various abstract or concrete data points.

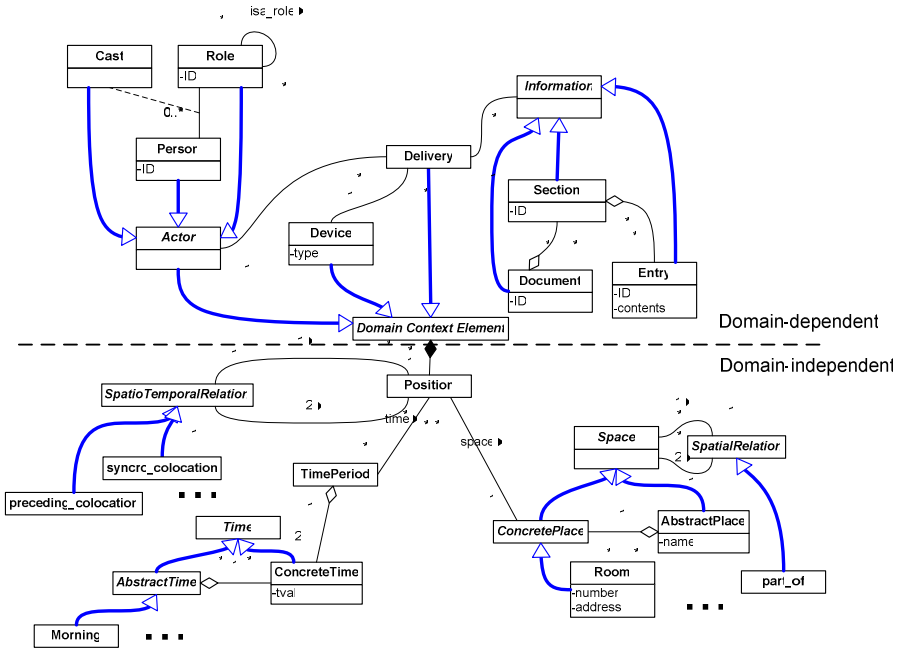

Figure 3 Context Ontology

The concept *space* is a generalisation of a *concrete* space or an *abstract* space. A concrete space is defined by an actual physical location, e.g., a room number at a certain address. An abstract space has a name (e.g., Victoria Hospital) and aggregates one or more concrete spaces. There exists (and extensible) set of spatial relations to relate spaces, e.g., *part of*.

5. Contextual Facts

The context ontology described in the previous section provides a formal model for representing contextual situations in a health care setting. Figure 4 shows an example for such a contextual situation: it is represented by a connected graph over instances of concepts and relations in the context ontology. Instances of domain-dependent ontology concepts are rendered with bold borders. The diagram shows two persons (David and Christina) casting as doctor and patient, respectively. Dr. David and Christina are located in room 321, which is also classified as an *office* (abstract place). Moreover, Dr. David interacts with a slit lamp exam section in a patient consult form delivered by a PC.

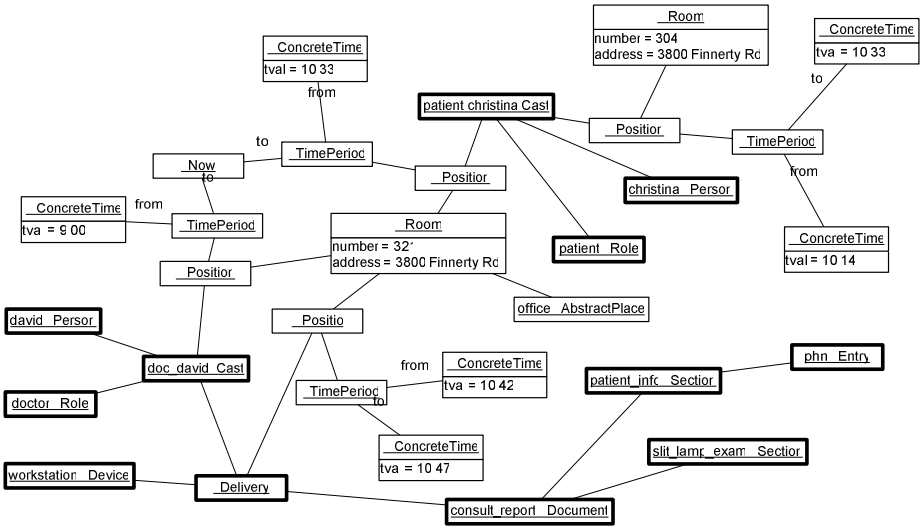

Figure 4 Actual Sample Context

In this paper, we do not focus on how to determine actual context situations from the environment, e.g., by means of smart sensors embedded in the environment. Rather, we are primarily interested in discussing a theoretical model for representing, storing and retrieving context for information delivery in health care. In principle, contextual facts, as described above, can be represented and stored in various ways. One possibility would be to use standard relational database technology. However, in this case, relations between instances of contextual concepts would be represented by means of value-based foreign keys, resulting in a large number of expensive join operations for context queries in large, highly associative fact bases. Since we expect

context queries to occur frequently in order to pick up situational changes, we decided that it was more appropriate to truly materialise the graph-based representation of contexts. One way to do so would be to develop and maintain appropriate data structures in object-oriented programming languages such as Java or C#. However, this solution would not support the execution of declarative context queries and look up. Therefore, we have chosen the graph-oriented, open-source database called GRAS [18] as a platform for the context fact base.

6. Context patterns and retrieval

The primary function of the CxMS is to associate background knowledge about predefined canonical context patterns with current facts about the actual situation at hand, and, in our case, present actors with the right information services. This section discusses the formalization of canonical context patterns, which will be used to retrieve and recognise canonical situations in the context fact base. Since context facts are represented as graphs as outlined in the previous section, we need to define a formalism expressing constraints on these graphs in order to serve as predicates for recognising canonical context patterns. Moreover, we are interested in *extending* the existing context fact base, based on the contexts recognised. Various formalisms for querying and rewriting graph structures exist, e.g., PROGRES [25] and GRAL [25], [14]. We have chosen programmed graph rewriting systems (PROGRES) as the platform for formalising context patterns, since PROGRES specifications are executable and integrate well with GRAS (the platform for our context fact base). PROGRES allows for the specification of so-called *graph productions*, which represent declarative rewriting rules on host graph structures.

Figure 5 shows an example for such a graph production, which has been defined to recognise a context *"Patient Consultation"*. A detailed introduction to the PROGRES language is beyond the scope of this paper. Therefore, we will only give an informal description of the example depicted here:

The left-hand side represent the *application condition* of the graph production. In the context of our CxMS, the left-hand side formalises the condition that has to be valid for a "*Patient Consultation*" context to be recognised. A successful recognition would imply that the context fact base contains a graph structure that is a suitable match for the left-hand side of the production rule. In this notation, normal boxes and edges are matched to nodes and edges in the fact graph. Additional *restrictions* can be added to nodes by means of doubled-edged arcs. For example, the restriction *valid(self.role.ID)="doctor"* attached to node `1 demands that this cast has the role of a "doctor". Analogously, the (user defined) restrictions *now* and *morning* demand that the cast is currently ongoing and that it is morning.[2] Another restriction demands that the doctor be located in an office.

[2] Note that we do not show the definition of restrictions *now* and *morning*. However, they can easily be defined based on a comparison of the time attribute *tval* of concept *ConcreteTime*.

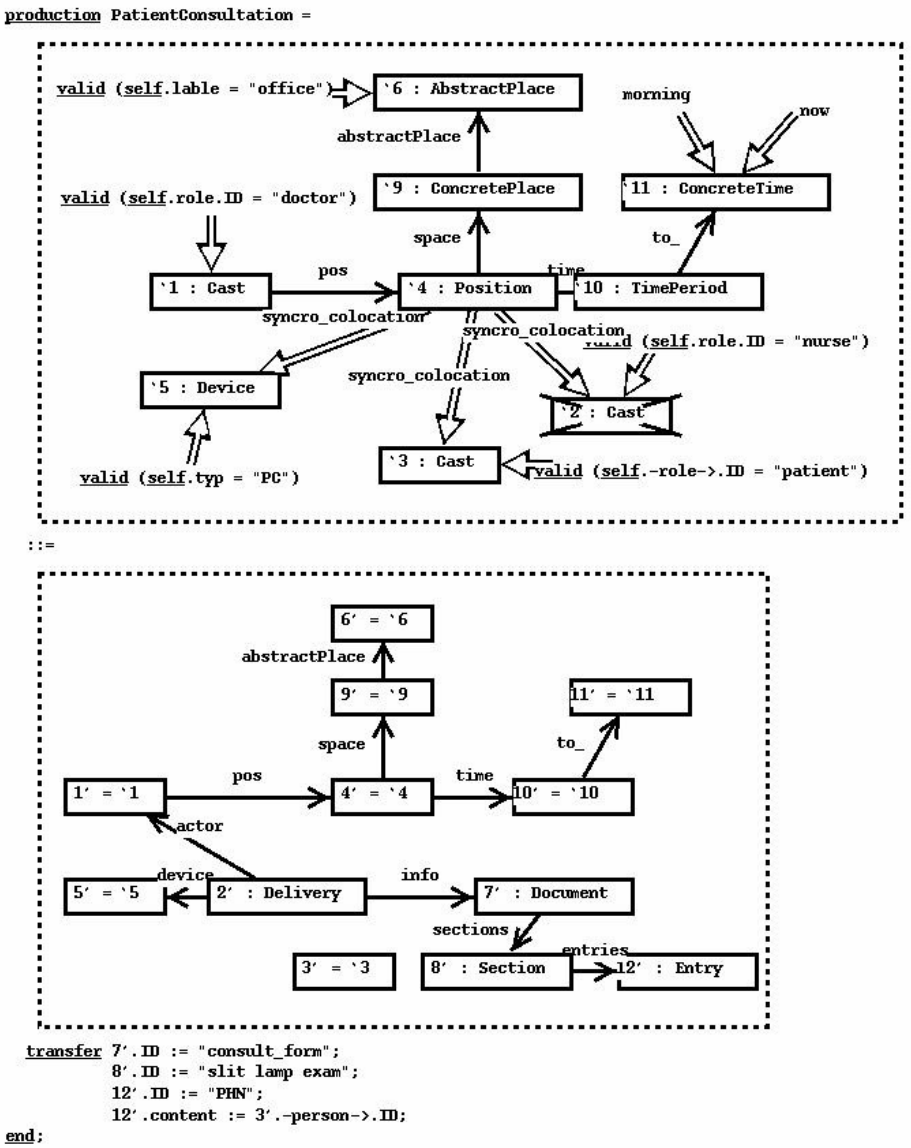

Figure 5. Example graph production Patient Consultation

Double-edged arcs between nodes are called *paths* in PROGRES. Paths can be used to declare more complex relationships among graph nodes. For example, the path *syncro_colocation* demands that a patient and a PC device must be in the same room at the same time. Its definition is presented in Figure 6: the path is defined between a node of type *Position* (`1 in Figure 6) and a node of type *DomainContextElement* (`2 in Figure 6). The graphical part of the definition of path *syncro_colocation* demands that the concrete place associated with the target DomainContextElement be a part of the place of the originating position. The textual

condition at the bottom of Figure 6 demands the temporal synchronicity of both positions.

Another notation used in the left-hand side of Figure 5 is a *negative node* (crossed-out rectangle), demanding that there may not be a nurse in the same location at the same time. Such a condition could be specified in order to protect the patient's privacy regarding sensitive health information.

If the left-hand side can successfully find a match in the context fact base, the production in Figure 5 is executed by replacing its left-hand side by its right-hand side. Nodes `1,`3,`4,`5,`6,`9,`10,`11 are preserved identically, whereas several new nodes are created in order to represent the information services to be delivered in the recognised context. Note that the *transfer* clause at the bottom of Figure 5 specifies that the information services delivered will include a patient consult form including a slit lamp exam. Moreover, they initialise the patient's identity (personal health number - PHN) in the appropriate entry of the form.

As a result of executing this graph production rule and the resulting change in the context fact base, the context view manager would render the appropriate information services (WebParts) on the doctor's office PC.

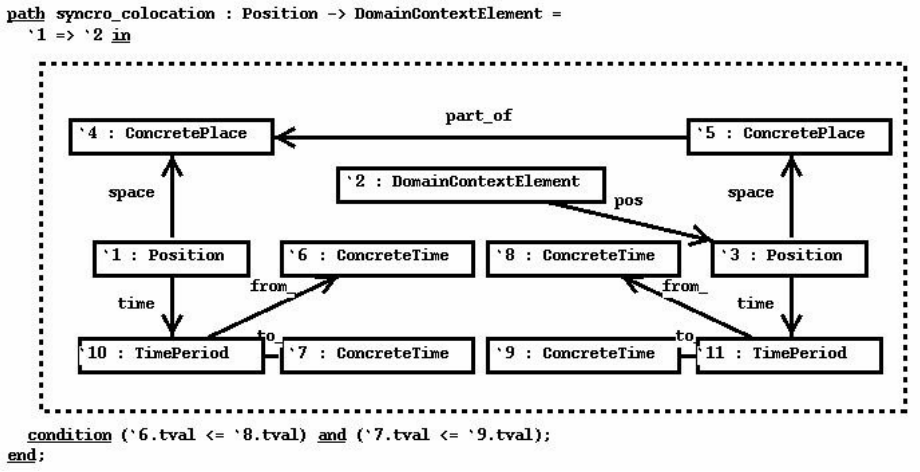

Figure 6. Definition of path *syncro_colocation*

References

[1] Aaltonen, A.: A Context Visualization Model for Wearable Computers. Proceedings of the Sixth International Symposium on Wearable Computers (ISWC 2002) (2002) 158-159

[2] Abowd, G., Atkeson, C., Hong, J., Long, S., Kooper, R., Pinkerton, M.: Cyberguide: a Mobile Context-aware Tour Guide. Wireless Networks, Volume 3, Issue 5 (1997) 421 - 433

[3] Anderston, J., Aydin, C.: Evaluating the Impact of Health Care Information Systems. Intern. J. Tech. Assess. In Health Care. 13(2) (1997) 380-393.

[4] Bardram, J.: Applications of Context-aware Computing in Hospital Work: Examples and Design Principles, Proceedings of the Symposium on Applied Computing (2004) 1574-1579
[5] Brown, P.: The Stick-e Document: a Framework for Creating Context-aware Applications. Proceedings of Electronic Publishing '96 (1996) 259–272
[6] Castro, P., Mani, M., Mathur, S., Muntz, R.: Managing Context for Internet Video Conferences: the Multimedia Internet Recorder and Archive. Proceedings of Multimedia and Computer Networks 2000 (2000)
[7] Chen, H., Finin, T., Anupam, J.: Semantic Web in a Pervasive Context Aware Architecture. Proceedings of Artificial Intelligence in Mobile System 2003
[8] Chen, H., Finin, T.: An Ontology for a Context Aware Pervasive Computing Environment. Proceedings of IJCAI workshop on ontologies and distributed systems, IJCAI'03 (2003).
[9] Chin, T.: Doctors Pull Plug on Paperless System. http://www.ama-assn.org/amednews/2003/02/17/bil20217.htm (2003)
[10] Chin, T.: Technology Valued, but Implementing it into Practice is Slow. http://www.ama-assn.org/amednews/2004/01/19/bisb0119.htm (2004)
[11] Davies, N., Cheverst, K., Friday, A., Mitchell, K.: Future Wireless Applications for a Networked City: Services for Visitors and Residents. IEEE Wireless Communications, Volume 9, Issue 1 (2002) 8- 16
[12] Davis, A., Moore, M., Storey, V.: Context-aware Communication for Severely Disabled Users. Proceedings of the ACM Conference on Universal Usability (2003) 106-111
[13] Dey, A.: Understanding and Using Context. Personal and Ubiquitous Computing, Vol. 5.1 (2001) 4-7
[14] Franzke, A.: GRAL: A Reference Manual. (1997).
[15] Hawryluk, M.: Government Pushes for Electronic Medical Record Standards. http://www.ama-assn.org/amednews/2004/02/09/gvsa0209.htm (2004)
[16] Holmquist, L., Gellersen, H., Kortuem, G., Schmidt, A., Strohbach, M., Antifakos, S., Michahelles, F., Schiele, B., Beigl, M., Maze, R.: Building Intelligent Environments with Smart-its. IEEE Computer Graphics and Applications, Volume 24, Issue1 (2004) 56- 64
[17] Junnarker, S.: Industrial Evolution: Intensive Care for Medical Data, Part 3. http://news.com.com/2030-1070-1001641.html (2003)
[18] Kiesel, N., A. Schuerr, et al.: GRAS, A Graph-Oriented (Software) Engineering Database System. Information Systems **20**(1) (1995) 21-52.
[19] Kindberg, T., Barton, J.: A Web-based Nomadic Computing System. Computer Networks 35(4) (2001) 443–456
[20] Kosara, R., Healey, C., Interrante, V., Laidlaw, D., Ware, C.: Visualization Viewpoints. IEEE Computer Graphics and Applications, Volume 23, Issue 4 (2003) 20- 25
[21] Ranganathan, A., Campbell, R.: An Infrastructure for Context-awareness Based on First Order Logic. Personal and Ubiquitous Computing, Volume 7, Issue 6 (2003) 353 - 364
[22] Salber, D., Dey, A., Abowd, G.: The Context Toolkit: Aiding the Development of Context-enabled Applications. *CHI* (1999) 434–441

[23] Schilit, B., Theimer, M.: Disseminating Active Map Information to Mobile Hosts. IEEE Network (1994) 22–32
[24] Schmidt, A., van Laerhoven, K.: How to Build Smart Appliances? IEEE Personal Communications, Volume8, Issue4 (2001) 66-71
[25] Schuerr, A.: Introduction to the Specification Language PROGRES. Building Thightly-Integrated (Software) Development Environments: The IPSEN Approach. M. Nagl, Springer. 1170 (1996) 248-279.
[26] Sohn, T., Dey, A.: iCAP: an Informal Tool for Interactive Prototyping of Context-aware Applications, Proceedings of Conference on Human Factors in Computing Systems (CHI '03) (2003) 974-975