A Trace-Based Framework for supporting Digital Object Memories

Lotfi S. SETTOUTI [a], Yannick PRIÉ [a], Damien CRAM [a], Pierre-Antoine CHAMPIN [a] and Alain MILLE [a]

[a] *LIRIS , University of Lyon, France, UMR CNRS 5205, Insa-Lyon, Université Lyon 1, Université Lyon 2, EC-Lyon*

Abstract. In this paper, we present a Trace Based framework for managing and transforming traces of observation and use of real life objects. Considering trace based systems as Digital Object Memories (DOMe), we describe how our framework can be used to manage DOMe, using trace models and transformations to raise the abstraction level of traces and infer useful knowledge. To demonstrate our approach, we present a simple fictional example of smart home where the use and the state of some of everyday objects are observed, and the resulting traces of such observations are exploited as DOMe providing useful services.

Keywords. Trace, Digital Object Memories, Trace Transformation, Trace-Based Reasoning

1. Introduction

The idea of creating context aware environments has long interested the research community, especially since the pervasive computing technologies (sensors, handheld or wearable devices, RFID and wireless technologies) have become cheaper and common. The focus has been twofold. At the lower level, attention has focused on getting data from sensors and instrumented objects and creating architectures and frameworks that support their integration [1,2]. At the upper levels, efforts have essentially focused on inferring context models from sensed information, with applications ranging in complexity from simple notifing/reacting systems (e.g. if $temperature > t$ then start air conditioner) to more complex systems that seek to build rich models and then reason over them (e.g. patient monitoring systems [3,4]). In particular, context aware real-world objects need mechanisms for reasoning about *whole histories* of sensed informations rather than reacting and reasoning only on recent information, requiring thus a real *Digital Object Memory* to provide intelligent services.

This workshop's call[1] defines the Digital Object Memories (DOMe) as comprised of *hardware and software components that physically and/or conceptually associate digital information with real-world objects in an application-independent manner. Such information can take many different forms (structured data and documents, pictures, audio/video streams, etc.) and originate from a variety of sources (automated processes,*

[1] http://www.dfki.de/dome-workshop/

sensors in the environment, users, etc.). If constantly updated, Digital Object Memories over time provide a meaningful record of an object's history and use.

As human memory, Digital Objects Memory is always changing; rarely does it stay the same over time and such changes are both *quantitative*, by adding continuously rough observations (at lower level), and *qualitative*, by aggregating and inferring new informations (often at more higher levels). Indeed, the management of DOMe needs a specific framework for dealing with observation at different levels, by providing the means to store and integrate informations emanating from several sources at lower level, together with mechanisms to abstract and infer interpretations about observed informations collected on real objects at several higher levels.

In this paper, DOMe are considered as *traces of observations captured and collected about real objects, associated with management and reasoning mechanisms for such traces*. Under such consideration, we define a trace-based framework as means of supporting the processing and transformations of traces. We will describe our approach with a fictional example not yet implemented, in order to precise and explain the introduced concepts and demonstrate implicitly the use of our framework to support DOMe applications.

The remainder of the paper is organized as follows. Section 2 gives an overview of th basic concepts of our approach and the architecture of our framework. Section 3 describes the fictional example that illustrates the models and transformations that can be applied. Our last section deals with discussion and future work.

2. Traces and Trace-Based System

The notion of trace as reification of the observation of the interaction of a human with a complex artefact (computer, vehicle) has long interested the SILEX[2] team. We have been studying various issues such as the modelling of traces of interactions or their manipulation for knowledge discovery within several application domains (activity reflection and analysis [5,6], decision support and reuse of traced experience [7,8,9,6]).

As a means to facilitate traces exploitation within our projects, we have defined the notion of Trace-Based System (TBS) as a kind of knowledge based system whose main source of knowledge is a set of continuously updated modelled traces describing users-systems interactions. We define the *trace model* as the schema or ontology describing the content of traces, and *modelled traces* as traces associated explicitly with their model. Assuming that trace-based systems use an abstract architecture as described in Figure 1, we will describe each component by specifying its use to support DOMe.

2.1. Trace-Based System Architecture

At the top of the general architecture of a TBS (Figure 1) is the tracing system, which collects the observed data from different input sources (log files, streamed actions, video records, interface events, sensor data, RFID signal, etc.). The *Tracing System* elaborates so called primary traces (often low level) from tracing sources.

A *Transformation System* allows to infer new trace elements by specifying transformation rules. Transformation rules provide a powerful mechanism to process traces, like

[2]http://liris.cnrs.fr/silex

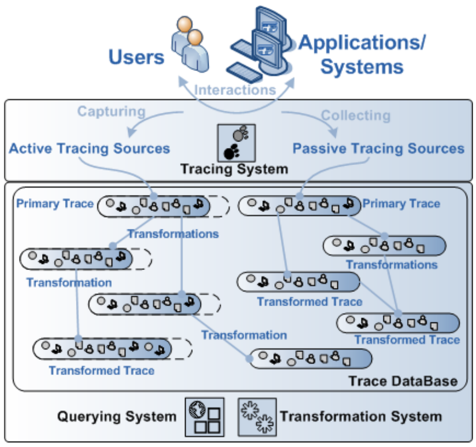

Figure 1. Trace-Based System Architecture

applying filters, rewriting and aggregating elements, computing elements attributes, etc. so as to produce so called transformed modelled traces that can be more easily reusable and exploitable in a given context than primary traces. Transformations can also be considered as semantic abstractions when associated with models (ontologies) of high level describing a specific task or activity. The Querying System enables the mining and extraction of episodes and patterns from the traces. All primary and transformed traces are stored in a persistant trace database.

To define and implement interoperable TBS systems, we have formally defined the above framework, specifying languages to model, query and transform interaction traces. Due to space limitations, we cannot describe these languages, but the interested reader can refer to [10]. We rather provide in the next section an example to illustrate that framework and to demonstrate how TBS can support DOMe manipulation and management.

3. Trace-Based System for Digital Object Memories

In this section we describe our framework through a fictional example in the context of real-world objects observation traces issued from common technologies such as sensors or Radio-Frequency IDentification (RFID). Both the collected traces and the mechanisms for their exploitation enact the role of digital memories for such objects.

3.1. Running Example

Consider the case of a smart home where three of everyday objects are equipped with observation capabilities. As showed in figure 2, let us consider a milk bottle and a baby bottle having both 1/ sensors that transmit bottle temperature measurements and 2/ RFID tags. Several RFID antennas are also integrated into the refrigerator, the microwave oven, within each room (baby room, kitchen, etc.). We assume that all RFID antennas within the smart home are able to read and recognize both milk and baby bottle RFID tags and to send this information to a computer system. We also assume that each transmitted bottle temperature comes with the temperature of the room containing its.

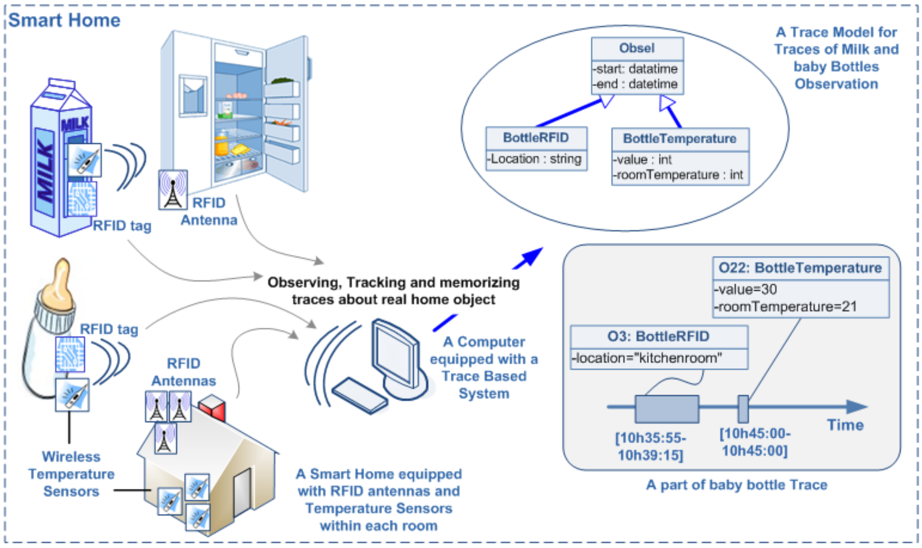

Figure 2. Smart home observation architecture with an example of resulted primary modelled trace

We assume that each object is associated to a trace or memory that stores information from the RFID antenna when an RFID tag activates it or the temperature sensor. Each piece of information about a bottle is associated to a time interval. In the case of temperature, the observation are instantaneous (no duration) delivered every 15 minutes, while RFID-obtained observations last as long as the antenna detects the RFID tag[3].

3.2. Modeling Traces

We defined a trace model as a kind of ontology describing the temporality, vocabulary and content of a trace. By considering traces as a support for digital object memories, our idea is to model the observation of a real instrumented object and to constitute the whole history of its life cycle at several abstraction levels, using transformations to "raise" low level memories to a higher level. Each trace model is therefore designed represent useful information at an adequate level, integrating streamed informations emanating from several sources.

For instance, in the upper right of Figure 2 the primary trace model for both bottles defines *Obsel*[4]*, BottleRFID, BottleTemperature* as the basic components of the observation ontology. Each concept corresponds to observations obtained from a sensor or RFID antenna concerns about a specific bottle (baby bottle or milk bottle in our example). A trace corresponding to that model for the baby bottle is illustrated in the lower right of Figure 2. It contains two observed elements. They corresponds to a observation of the bottle being in kitchen during 4'20" at 10h35, and an observation of its temperature being 30°C at 19h45 while the temperature in the room is 21°C.

[3] The preprocessing required to obtain this kind of information could be performed inside the TBS, but we ignore it for the sake of simplicity.

[4]"Obsel" is contraction of "OBServed ELement"

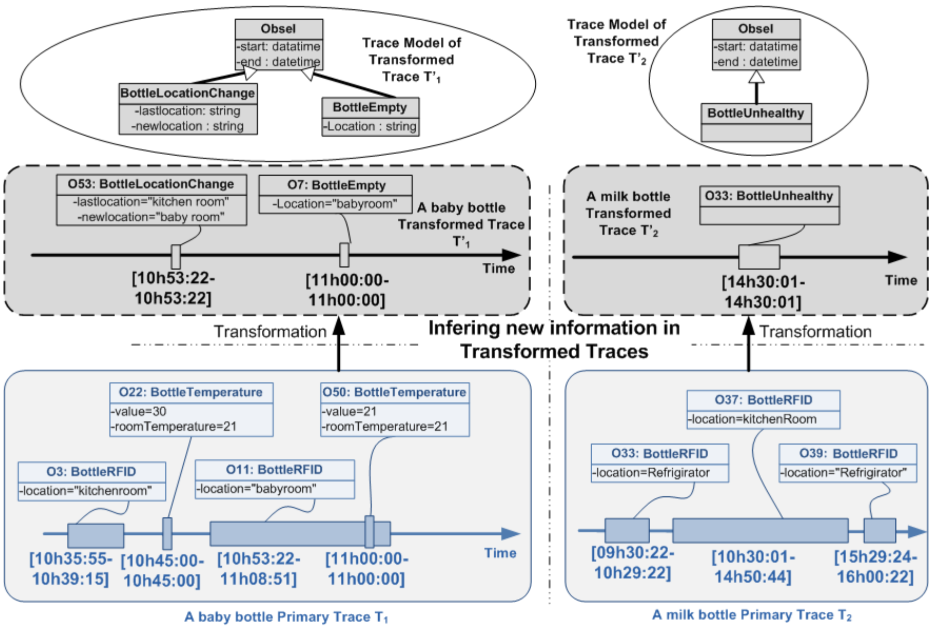

Figure 3. Trace Transformations to improve DOMe

3.3. Querying and Transforming Traces to improve Digital Object Memories

TBS allows the expression of pattern-extraction based queries and transfromation rules to infer and detect complex situations. TBS supports rules as an abstraction and reasoning mechanism for traces, with the same motivation and benefits of rules in deductive and knowledge based systems [10]. Based on the primary trace model and a defined model for the transformed trace, a transformation allows to populate a new transformed trace by specifying 1/ the pattern (rule body) describing which elements will be matched to be transformed and 2/ the template (rule head) to generate and infer the new observed elements from the ones retrieved by the patterns. For instance, Figure 3 illustrates how such a transformation, producing the transformed trace T_1' from the primary trace T_1, can simply infer higher level information such as *BottleLocationChange* when two successive *BottleRFIDs* within T_1 do not have the same location. More complex tranformation rules can infer a new observation such as *BottleEmpty* when two successive *BottleTemperatures* show a quick convergence of the bottle temperature towards the room temperature. This kind of information can be useful in the memory of baby bottle indicating that the baby bottle is empty and it is left in the baby room.

Let us now consider the milk bottle memory: trace T_2 in Figure 3 shows how the milk bottle is detected in the refrigerator, then the kitchen then the refrigerator again. Based on a transformation of trace T_2, the TBS can infer within T_2' that, since the milk has been left out of the refrigerator for more than four hours, it has probably become unhealthy. Such information can be very useful for the person preparing the baby bottle, especially if s/he is not the one who put the milk back in the refrigerator at 15h29. The TBS can send an email or SMS warning message to home members, indicating that the milk bottle is possibly inappropriate for use in baby bottle preparation.

4. Discussion and future work

In this paper, we gave a brief overview of Trace Based Systems (TBS), a framework for managing and transforming modelled traces about observation and use of real life objects. We have demonstrated, with a simple fictional example, how that framework can be used to manage DOMe, using transformations to raise the abstraction level of traces and infer useful knowledge. Conversely to context-aware application frameworks (as [11]) that use context abstraction based on XML technologies, we use ontologies with precise semantics (see [10]) to express more complex context specification and reasoning mechanisms.

That approach is actually being developed and experimented in several contexts: e-Learning personalisation, personal and corporate knowledge management. Beyond these software-centred user-system interactions, we also used our framework in the study of vehicle driving behaviours, and are developing the use of other sources of observations (eye tracking, sensors, RFID, etc.), especially in the context of accessibility for the disabled.

However, our framework has been designed and examplified to manage several DOMes within a centralised architecture. A more in-depth investigation to applying our framework to open and distributed environment is currently on our agenda.

References

[1] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", in *ACM International Conference on Mobile Computing and Networking (MOBICOM'00*, 2000, pp. 56–67.

[2] Levent Gurgen, Claudia Roncancio, Cyril Labbé, André Bottaro, and Vincent Olive, "Sstreamware: a service oriented middleware for heterogeneous sensor data management", in *ICPS '08: Proceedings of the 5th international conference on Pervasive services*, New York, NY, USA, 2008, pp. 121–130, ACM.

[3] Sheetal Agarwal, Anupam Joshi, Tim Finin, Yelena Yesha, and Tim Ganous, "A pervasive computing system for the operating room of the future", *Mob. Netw. Appl.*, vol. 12, no. 2-3, pp. 215–228, 2007.

[4] EMJ Koski, T. Sukuvaara, A. Mäkivirta, and A. Kari, "A knowledge-based alarm system for monitoring cardiac operated patients-assessment of clinical performance", *Journal of Clinical Monitoring and Computing*, vol. 11, no. 2, pp. 79–83, 1995.

[5] Olivier Georgeon, Matthias J. Henning, Thierry Bellet, and Alain Mille, "Creating Cognitive Models from Activity Analysis: A Knowledge Engineering Approach to Car Driver Modeling", in *International Conference on Cognitive Modeling*. July 2007, pp. 43–48, Taylor & Francis.

[6] Damien Cram, Béatrice Fuchs, Yannick Prié, and Alain Mille, "An approach to user-centric context-aware assistance based on interaction traces", in *MRC 2008, Modeling and Reasoning in Context*, 2008.

[7] Pierre-Antoine Champin, Yannick Prié, and Alain Mille, "Musette: Modelling uses and tasks for tracing experience", in *workshop From structured cases to unstructured problem solving episodes - WS 5 of ICCBR'03*, Béatrice Fuchs and Alain Mille, Eds., Trondheim (NO), 2003, pp. 279–286.

[8] Julien Laflaquière, Lotfi Sofiane Settouti, Yannick Prié, and Alain Mille, "A trace-based System Framework for Experience Management and Engineering", in *Second International Workshop on Experience Management and Engineering (EME 2006) in conjunction with KES2006*, Oct. 2006.

[9] Alain Mille, "From case-based reasoning to traces-based reasoning", *Annual Reviews in Control*, vol. 30, no. 2, pp. 223–232, Oct. 2006, Journal of IFAC.

[10] Lotfi Sofiane Settouti, Yannick Prié, Pierre-Antoine Champin, Jean-Charles Marty, and Alain Mille, "A Trace-Based Systems Framework : Models, Languages and Semantics", technical report, LIRIS , University of Lyon, France, UMR CNRS 5205, Université Lyon 1, 2009, http://hal.inria.fr/inria-00363260/en/.

[11] Anind Kumar Dey, *Providing architectural support for building context-aware applications*, PhD thesis, Atlanta, GA, USA, 2000, Director-Abowd, Gregory D.