

# Towards a Standards-Based Autonomic Context Management System<sup>\*</sup>

Jadwiga Indulska, Karen Henriksen, and Peizhao Hu

School of Information Technology and Electrical Engineering,  
The University of Queensland  
and

National ICT Australia (NICTA)

{jaga, peizhao}@itee.uq.edu.au, Karen.Henriksen@nicta.com.au

**Abstract.** Pervasive computing applications must be sufficiently autonomous to adapt their behaviour to changes in computing resources and user requirements. This capability is known as context-awareness. In some cases, context-aware applications must be implemented as autonomic systems which are capable of dynamically discovering and replacing context sources (sensors) at run-time. Unlike other types of application autonomy, this kind of dynamic reconfiguration has not been sufficiently investigated yet by the research community. However, application-level context models are becoming common, in order to ease programming of context-aware applications and support evolution by decoupling applications from context sources. We can leverage these context models to develop general (i.e., application-independent) solutions for dynamic, run-time discovery of context sources (i.e., context management). This paper presents a model and architecture for a reconfigurable context management system that supports interoperability by building on emerging standards for sensor description and classification.

## 1 Introduction

There is a growing body of research on the use of context-awareness as a technique for developing applications that are flexible, adaptable, and capable of acting autonomously on behalf of users. This research addresses context modelling, management of context information, techniques for describing and implementing adaptive behaviour, and architectural issues. However, further research is needed before context-aware applications can be said to be truly autonomic.

The research so far focuses mostly on one aspect of autonomy: evaluation of context information by context-aware applications to make decisions about necessary adaptations to the current user context/situations. While numerous architectures and context management systems exist that define how sensor data

---

<sup>\*</sup> National ICT Australia is funded by the Australian Government's Department of Communications, Information Technology, and the Arts; the Australian Research Council through Backing Australia's Ability and the ICT Research Centre of Excellence programs; and the Queensland Government

used in context-aware applications can be gathered, processed and evaluated, the solutions developed so far address a reasonably static environment. In particular, they assume that a context-aware application is supported by a pre-defined set of information sources, which typically take the form of physical and logical sensors. They do not consider the types of autonomic behaviour required to (i) support mobile users who move between environments or administrative domains, or (ii) overcome problems such as sensor failures or disconnections.

Compelling examples of applications that require a high degree of autonomic behaviour include vehicles that need to fuse data from on-board sensors with sensors encountered in the environment (e.g., military unmanned vehicles and intelligent wheelchairs able to discover sensors in the infrastructure to adapt their path), and emergency response software that opportunistically gathers information from sensors located in the vicinity of an emergency. In this paper, we address the challenges of highly autonomic context-aware applications by proposing a model for autonomic, run-time reconfiguration of context management systems. To support both dynamic discovery of context information and interoperability between different context management domains, our model incorporates newly emerging standards for sensor description and classification.

The structure of the paper is as follows. Section 2 presents two application scenarios that motivate the work presented in this paper. Section 3 presents a context model for one of the scenarios, which we use in our discussions and examples in the remainder of the paper. Section 4 reviews existing approaches to context management as well as emerging standards for sensor description and classification. Section 5 describes our standards-based model for sensor description, while Section 6 presents the architecture of our reconfigurable context management system and illustrates the roles of context models and sensor descriptions in supporting dynamic discovery of sensors. Section 7 illustrates how sensor observations can be mapped into application-level context models, and Section 8 summarises the contributions of the paper.

## 2 Scenarios

Context-aware applications rely on evaluation of context information to make decisions about necessary adaptations to the current context/situations. This information is most commonly gathered from sensors; therefore, numerous architectures and context management systems have been developed to support the gathering, processing and evaluation of sensor data. However, these are unsuitable for applications that require a high degree of autonomic behaviour because these applications (i) operate in frequently changing environments with dynamically changing context sources and (ii) cannot rely on explicit setup or re-configuration of context sources by humans (either users or administrators). In this section, we present two scenarios that illustrate such applications and their requirements with respect to the sensing infrastructure.

**Emergency Services Scenario.** A context-aware application providing surveillance of critical infrastructure is supported by a network of sensors, includ-

ing cameras, audio sensors, temperature and light sensors, and weather stations (detecting fog, rain, and so on). The surveillance application provides sophisticated processing of sensor information including face, scene and number plate recognition for detection of abnormal events. Part of the sensor network is destroyed in a physical attack. Emergency services arrive in vehicles equipped with cameras, audio sensors and temperature sensors. Based on the application context model, which describes the types of context information required by the application, a discovery protocol identifies new vehicle-based sensors that can be used by the surveillance application in place of the lost sensors (taking into account sensor types, capabilities and locations). The application is dynamically reconfigured to use the newly discovered sensors.

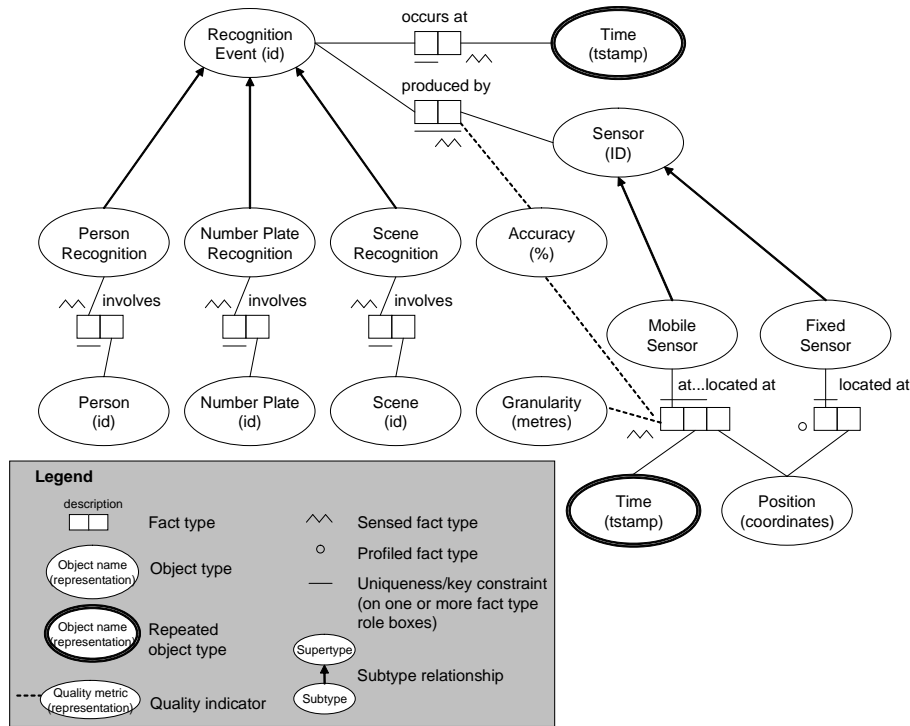
**Mobile Phone Scenario.** A next-generation mobile phone adapts its configuration according to the user's context. In order to support this behaviour without on-board sensors, a small context gathering utility is installed on the phone to collect relevant context information from sensors encountered within the environment. The phone is capable of determining its location (within a few metres accuracy) using triangulation, and it uses this knowledge to support discovery of information supplied by nearby sensors. The phone takes advantage of information from a wide variety of sensor types, including the lighting level (to appropriately adjust backlight settings) and noise level (to ensure that the ring volume is audible). In addition, it fuses the information from all of the available sensors to infer the user's current activity: for example, it infers that the user is probably sleeping when the lights are off and there is no sound or movement, and switches automatically to silent mode.

### 3 Context Modelling Approach

Context-aware applications have traditionally been developed using one of the following three approaches:

1. each application directly queries sensors and processes the sensor data to make decisions about how to adapt (*no application-level context model*);
2. applications are built using shared context processing infrastructure/toolkits that assist with gathering and processing data (*implicit context model*); or
3. applications have their own well-defined context models and use a shared context management infrastructure to populate the models at run-time using sensors and other context sources (*explicit context model*).

The third approach is the most appropriate for applications that require highly autonomic context management. Although a variety of context models have been proposed for context-aware systems [1], some are more suitable than others. The most appropriate models for autonomic context management are those that define not only the types of information required by the application, but also metadata that can be used in binding the model to suitable sensors (for example, quality metadata and classification of information types into sensed and non-sensed types). Our previously developed fact-based context modelling



**Fig. 1.** Context model for an emergency response application

approach [2, 3] meets this requirement, and is therefore used as the basis for the work presented in this paper. However, it should be noted that our solution for autonomic context management can be used with any fact-based model.

To illustrate our context modelling notation, we present an example context model in Fig. 1 for the emergency services scenario described in Section 2. The context model describes three types of events that can be recognised by various types of sensors: face recognition events, number plate recognition events and scene recognition events. Face recognition events are associated with a person ID derived by matching to a database of known people. Similarly, number plate recognition events are associated with plate numbers, and scene recognition events with scene identifiers.

Each recognition event is associated with a timestamp and one or more sensors. Sensors in turn have locations (a single location value in the case of fixed sensors, and multiple location readings with associated timestamps in the case of mobile sensors). Each event-sensor pair (represented by the “produced by” fact type) can also have an accuracy measure, showing the rate at which the sensor correctly detects events of the given type. Similarly, location readings can be associated with both accuracy and granularity measures.

## 4 Related Work

As mentioned in the previous sections, the research community working on context-aware applications has developed a variety of software infrastructures and programming toolkits which can be used by applications to obtain context information from sensors. These are comprehensively discussed in one of our recent papers [4]; therefore, we mention only the most relevant solutions here. Arguably the most well known solution is Dey et al.'s Context Toolkit [5]. However, a crucial shortcoming of this solution (and most others in the field) is that it produces a reasonably tight coupling between applications, sensors, and intermediaries responsible for interpreting low-level sensor data. Remote communication between the components responsible for sensing and processing context data is carried over HTTP, which requires pre-configuration of IP addresses or explicit discovery using *discoverer* components. Although pre-configuration and explicit discovery of components may be acceptable in reasonably static environments, it is not appropriate for environments in which mobility, failures or disconnections are common. A preferable solution for these latter environments would be to allow applications to specify their requirements for context information in high-level terms, and to leave it up to the infrastructure/middleware to dynamically re-discover and re-bind sensors as required.

One of the most advanced solutions produced by the context-awareness community is Solar [6]. While this solution does not entirely free application developers from specifying how context information is derived, it does remove the task of discovering components from the application source code. The developer specifies the derivation of context information in the form of operator graphs, which include sources, sinks and channels. In this model, sensors are represented as sources, and application components as sinks. Intermediate components responsible for interpreting or otherwise processing the data act as both sources and sinks. At run-time, operator graphs are instantiated and managed by the Solar platform. The platform is capable of handling both mobility and component failures. However, it still requires the sensor configuration to be well specified in advance in the operator graph, rather than allowing for truly dynamic discovery and reconfiguration. It is not possible to substitute previously unknown types or configurations of sensors that are capable of providing the required information.

The sensor network community has addressed dynamic discovery and reconfiguration to a greater extent than the context-awareness community. However, this community predominantly focuses on low-level issues, such as networking protocols for wireless ad-hoc sensor networks [7] and protocols that provide low power [8] or energy-aware [9] operation. Similarly, the discovery protocols and query languages developed by the sensor network community are low-level, rather than application-focused. That is, they address problems such as finding neighbouring sensors and supporting database-like queries over sensor networks [10], rather than enabling high-level requests that specify only the nature of the required context information and desired information quality, without reference to particular sensors or sensor types.

Separate work is currently underway to develop standards for producing comprehensive descriptions of sensors. The results of this work include the Sensor Model Language (SensorML)<sup>1</sup>, which is being developed as part of the Sensor Web Enablement (SWE) initiative with the goal of enabling sensors to become accessible via the Web, and the IEEE 1451 family of standards<sup>2</sup>, which describes transducer (i.e., sensor or actuator) interfaces to enable interoperability. The goals of these two standards efforts are quite different: the SWE initiative is mainly concerned with describing sensor capabilities and observations (including geolocation of observations), while IEEE 1451 concentrates on the static characteristics of the sensor hardware. However, both families of standards can be used to help support dynamic discovery of sensors and/or observations. In particular, the SWE recommendations enable metadata-based discovery using standard semantic Web search tools and methods. To date, however, very little work has been done on sensor discovery and management using these standards (although work on sensor and observation discovery using Sensor Registries is planned by members of the SWE initiative). In this paper, we consider these issues in light of the requirements of autonomic context-aware systems that are based on well defined context models. In particular, we look at how SWE and IEEE 1451 descriptions can be combined and augmented to support autonomic discovery and management of sensors. The following section addresses sensor description using the two families of standards in more detail.

## 5 Sensor Descriptions

To support both wide adoption and interoperability, the sensor descriptions that support dynamic sensor discovery, identification and configuration should be based, to the extent possible, on standard descriptions and classifications of sensors. This is the main reason that our approach for describing sensors is based on the two families of emerging standards which are briefly discussed in this section.

**IEEE 1451 Family of Standards.** IEEE 1451 comprises a family of open standards defining common interfaces for transducers (sensors and actuators) to communicate with other components. Three standards from the family, 1451.0, 1451.2 and 1451.4, specify TEDS (Transducer Electronic Data Sheet), which contains detailed information that can be used for sensor identification and self-configuration, ranging from the sensor model number to sensing capabilities.

IEEE P1451.0 is a preliminary (with prefix “P”) standard, which aims to define a set of common commands, common operations and TEDS for IEEE 1451 smart transducers. IEEE 1451.2 offers network independence by introducing Network Capable Applications Processors (NCAP), together with an extensive set of stand-alone TEDS for a wide range of transducers. IEEE 1451.4 defines the Mixed-Mode Interface to enable analog and digital signals to share the same wires, and redefines a subset of 1451.2 TEDS to minimise the size of non-volatile

---

<sup>1</sup> <http://vast.uah.edu/SensorML/>

<sup>2</sup> <http://ieee1451.nist.gov/>

memory used by TEDS. Limited configuration information is encoded according to assigned templates, and stored either in Electronic Erasable Programmable ROM or in a Virtual TEDS<sup>3</sup> file that is accessible through the Internet.

We base our approach on IEEE 1451.4 as it provides a flexible, memory-lean TEDS model to suit transducers with varying capabilities, including small devices such as microphones, thermocouples and accelerometers.

**Open Geospatial Consortium Recommendations.** The Sensor Web Enablement initiative is an ongoing activity carried out by the Open Geospatial Consortium (OGC)<sup>4</sup> with the aim of allowing sensors and sensor data to become discoverable and accessible in real time over the Web. Two of the main products of the initiative so far have been the SensorML and Observations & Measurements (O&M) recommendations. These deal with the representation of sensor descriptions and observations, respectively.

SensorML and O&M can each be used to support aspects of autonomic context management, as we show in the remainder of this paper. SensorML descriptions can be used for discovery of context sensors (as discussed in Sections 5 and 6), while O&M can be used to support the mapping of sensor observations into application-level context models (as discussed in Section 7).

SensorML models sensors as processes with inputs, outputs, parameters and methods. SensorML processes can be composed into process chains in order to model composite sensor systems. Four properties of sensors in the model that are relevant for reconfiguration are *sml:identification*, *sml:classification*, *sml:capabilities*, and *sml:characteristics*. In a similar way to the IEEE 1451 TEDS, *sml:identification* and *sml:classification* are used to assist sensor discovery and “plug-n-play” functionality: *sml:identification* provides information such as the manufacturer ID, model number and serial number, while *sml:classification* provides information concerning sensor types, applications, and supported sensing phenomena. *sml:capabilities* and *sml:characteristics* provide detailed technical specifications which can be used for autonomic sensor reconfiguration.

The O&M model defines a number of terms used for describing sensed observations and relationships between them. An observation is modelled as an event with a timestamp and a value describing the sensed phenomenon. The *locator* property is utilised to define the location of the observation, while a *procedure* described using SensorML defines the sensor or instrument used to capture the observations. Finally, related observations (such as a sequence of observations taken at regular intervals) can be grouped using observation arrays. An example observation for a “FaceRecognition” phenomenon is shown later in Section 7.

**Hybrid Sensor Description Model.** We envisage that in the future, most sensors will provide either embedded or virtual TEDS descriptions; therefore, we describe sensors by integrating SensorML and TEDS. That is, we base the part of the SensorML sensor description related to sensor identification, classification, reconfiguration and calibration on the IEEE 1451 standard. The advantage of

---

<sup>3</sup> <http://www.ni.com/teds>

<sup>4</sup> <http://www.opengeospatial.org>

Name	Value
SensorML version 1.0	
Sensor	id = SSC-CD53V Camera
i Identification	
longName (Term)	Sony SSC-CD53V Dome Camera
shortName (Term)	Sony CD53V
modelName (Term)	SSC-CD53V
manufacturer (Term)	Sony
serialNumber (Term)	unknown
cameraType (Term)	Dome
i Classification	
sensorType (Term)	camera
intendedApplication (Term)	motionDetection
intendedApplication (Term)	surveillance
phenomenon (Term)	image
phenomenon (Term)	video
phenomenon (Term)	Motion
phenomenon (Term)	FaceRecognition
phenomenon (Term)	NumberPlateRecognition
phenomenon (Term)	SceneRecognition
i Capabilities	{Comment: Details Deleted}
i Characteristics	
Inputs	
Outputs	
Parameters	

**Fig. 2.** Camera SensorML example (shown as a screenshot from the SensorML editor from <http://vast.uah.edu/SensorML/>, rather than an XML document, for brevity)

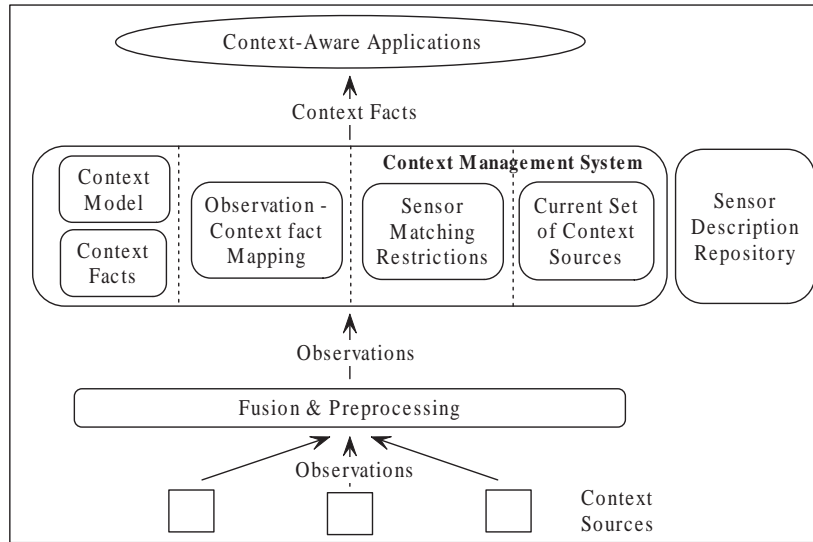
this approach is that this part of the SensorML description can be generated automatically from the TEDS description.

Sensors can be either physical (e.g., temperature sensors, GPS devices) or logical, which are often physical sensors enhanced with additional software to support more sophisticated types of sensing phenomena than the physical sensor alone. An example of a logical sensor is the camera described in Fig. 2. The camera as a physical sensor provides a stream of images, whereas the smart camera required in our emergency services scenario is augmented with face, scene and number plate recognition software. This is indicated in the camera description in Fig. 2 by the list of sensing phenomena this particular sensor/device provides.

## 6 Reconfiguration of Context Sources

The role of context management systems is to acquire and store context information, typically in the form of context facts, and to disseminate this information to applications through querying and/or notification based on subscriptions.

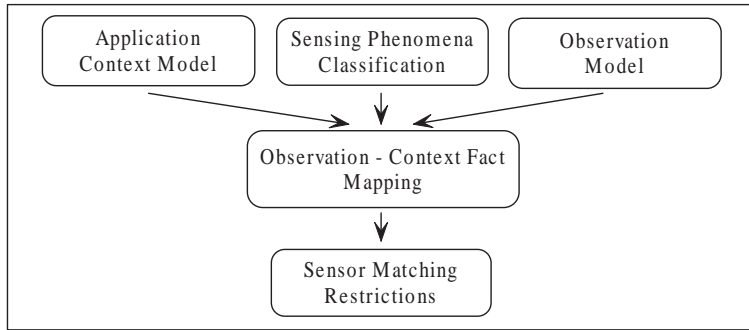
Figure 3 illustrates a simplified architecture of a context-aware system, in which only the context management system is shown at the middleware level. The left side of the figure (Context Model and Context Facts) represents information stored in most current context management systems, including our own previously developed system [4]. However, we argue that in order to allow run-



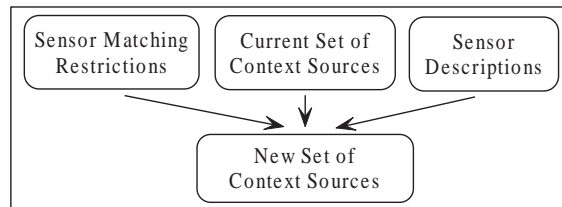
**Fig. 3.** Architecture

time discovery and replacement of context sources, some additional information and functionality is required in context management systems. This includes:

1. *Sensor Description Repositories*, to support discovery of sensors. As we have already shown in Section 5, sensor descriptions can be based on standards to support interoperability in pervasive systems. We use a combination of TEDS and SensorML, as TEDS supports dynamic discovery and identification of sensor instances, while SensorML can describe additional information needed in sensor matching and management.
2. *Current Set of Context Sources*, to support run-time replacement of sensors. The context management system requires knowledge of current sensor bindings in order to facilitate their replacement in the case of sensor failures or user mobility.
3. *Observation - Context Fact Mapping*, to support the integration of sensor observations from arbitrary sensors into the application-level context model, which represents context information in terms of facts. This mapping can be specified by the designer of the context model in terms of metadata that extends the model. We propose an approach in which the mapping assumes that inputs are in the observation format specified by the OGC's O&M recommendation, in order to facilitate interoperability across sensor types and context management systems.
4. *Sensor Matching Restrictions*, to support discovery of new sensors based on context fact types defined for the application. As shown in Fig. 4, the restrictions should be created by the application designer based on the *Observation*



**Fig. 4.** Design process



**Fig. 5.** Re-organisation

- *Context Fact Mapping.* The restrictions should stipulate the required sensing phenomena (e.g., “FaceRecognition”), as well as application-dependent restrictions (e.g., sensors must be within a well-defined geographical region or in close proximity to a given object).

Context management systems that provide these features can support dynamic re-organisation as follows: based on both the description of the current sources (including the sources that must be replaced) and the sensor matching restrictions, the sensor description repository can be searched for appropriate sensors. This is shown in Fig. 5.

## 7 Mapping Observations to a Context Model

As discussed in the previous section, reconfigurable context management systems not only need to be able to dynamically discover appropriate sensors with which to populate application-level context models, but also to map observations reported by the chosen sensors to the representations used by the models (in our case, context facts). The mapping from sensor observations to high-level context information can be specified at design-time in terms of mapping descriptions that extend the context model. Although the mapping can potentially be expressed

```

<gml:Observation>
  <gml:timestamp>
    <gml:TimeInstant>
      <gml:timePosition>2005-09-27T23:32:54</gml:timePosition>
    </gml:TimeInstant>
  </gml:timestamp>
  <om:using xlink:href="http://equipment.itee.uq.edu.au/foyer/camera03" />
  <gml:resultOf>
    <gml:Category>P343989961</gml:Category>
  </gml:resultOf>
  <om:observable xlink:href="phenomena.xml#FaceRecognition"/>
  <om:quality method="quality.xml#error">
    <om>Error uom="units.xml#percent">5</om>Error>
  </om:quality>
</gml:Observation>

```

**Fig. 6.** Face recognition observation

in any format, basing it on standards promotes interoperability. Therefore, we base our mapping on the O&M recommendation.

In the remainder of this section, we illustrate an example mapping related to the emergency services scenario and the context model shown in Fig. 1. As discussed in Section 3, the context model captures three types of recognition events (face, number plate and scene recognition). Therefore, the sensor matching restrictions for this model (described briefly in the previous section), should stipulate the required phenomenon types (“FaceRecognition”, “SceneRecognition” and “NumberPlateRecognition”), as well as restrictions on the geographical area in which the sensors are located. We don’t show these restrictions, as they vary from one emergency to another and therefore need to be generated at run-time.

Each of the three observation types requires its own mapping to one or more facts. We focus on the mapping of observations for the “FaceRecognition” phenomenon by way of example. Figure 6 shows a face recognition observation specified using O&M. The observation, produced by a camera described by the SensorML document located at “http://equipment.itee.uq.edu.au/foyer/camera03”, represents an event in which the camera recognises a person with ID P343989961 with a 5% likelihood of error. This observation can be mapped to the following facts, where fact types are shown in brackets:

- “23434 involves P343989961” [PersonRecognition involves Person]
- “23434 occurs at 2005-09-27T23:32:54” [RecognitionEvent occurs at Time]
- “23434 produced by http://equipment.itee.uq.edu.au/foyer/camera03”, accuracy = 0.95 [RecognitionEvent produced by Sensor]

The mapping of the observation values to the above facts is performed on the basis of fact templates that select values from observations using the XML Query language (XQuery)<sup>5</sup>, and then carry out any necessary transformations.

<sup>5</sup> <http://www.w3.org/XML/Query/>

## 8 Conclusions

In this paper we presented a model and architecture for autonomic context management systems which can discover and replace sources of context information at run-time. This work extends our previous context management system, developed as part of our infrastructure for context-aware applications [4].

The proposed reconfigurable context management system is a general solution suitable for highly autonomic, context-aware applications that use context models specified in terms of high-level facts. To create sensor descriptions that support dynamic sensor discovery we combined SensorML and IEEE 1451. In addition, to support the integration of observations from arbitrary sensors into application-level context models at run-time, we developed a mapping technique based on the OGC's O&M recommendation for representing sensor observations. The use of emerging standards in our solution leverages the fact that future sensors will provide standard descriptions, and promotes interoperability between context management systems.

## References

1. Strang, T., Linnhoff-Popien, C.: A context modeling survey. In: UbiComp 1st International Workshop on Advanced Context Modelling, Reasoning and Management, Nottingham (2004)
2. Henricksen, K., Indulska, J., Rakotonirainy, A.: Modeling context information in pervasive computing systems. In: 1st International Conference on Pervasive Computing (Pervasive). Volume 2414 of Lecture Notes in Computer Science., Springer (2002) 167–180
3. Henricksen, K., Indulska, J.: Developing context-aware pervasive computing applications: Models and approach. *Journal of Pervasive and Mobile Computing* **2** (2006) 37–64
4. Henricksen, K., Indulska, J., McFadden, T., Balasubramaniam, S.: Middleware for distributed context-aware systems. In: International Symposium on Distributed Objects and Applications (DOA). Volume 3760 of Lecture Notes in Computer Science., Springer (2005) 846–863
5. 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* **16** (2001) 97–166
6. Chen, G., Li, M., Kotz, D.: Design and implementation of a large-scale context fusion network. In: 1st Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous), IEEE Computer Society (2004) 246–255
7. Lim, A.: Distributed services for information dissemination in self-organizing sensor networks. *Journal of the Franklin Institute* **338** (2001) 707–727
8. Polastre, J., Hill, J., Culler, D.: Versatile low power media access for wireless sensor networks. In: 2nd International Conference on Embedded Networked Sensor Systems, Baltimore (2004) 95–107
9. Shah, R.C., Rabaey, J.M.: Energy aware routing for low energy ad hoc sensor networks. In: Wireless Communications and Networking Conference (WCNC). Volume 1. (2002) 350–355
10. Yao, Y., Gehrke, J.: Query processing for sensor networks. In: 1st Biennial Conference on Innovative Data Systems Research (CIDR), Asilomar (2003)