

Sensor Standards: Overview and Experiences

Peizhao Hu^{1 2}, Ricky Robinson², Jadwiga Indulska^{1 2}

¹ *The University of Queensland*

Brisbane, QLD 4072, Australia

{peizhao, jaga}@itee.uq.edu.au

² *NICTA*

300 Adelaide Street, Brisbane, QLD 4000, Australia

Ricky.Robinson@nicta.com.au

Abstract

Sensors are swiftly finding their way into real-world applications, from farming to factory floor monitoring. In parallel with this deployment, several standards are being developed to extend greater interoperability between sensor systems. In this paper, we explore a subset of the various sensor standards, and relate our experiences in attempting to integrate some of these standards for the purposes of developing an autonomic context manager.

1. INTRODUCTION

Sensors are increasingly proposed in the literature as a solution to a wide number of problems, and are slowly being deployed in some real-world applications [5]. Typically they are deployed for the purposes of gathering data such as temperature, light intensity and object motion in environmental research and animal behaviour studies. However, in the emerging paradigm of pervasive computing, in which users are mobile and decoupled from the devices they use, sensors can play an important role in enabling application adaptation. The information gathered from sensors is used by so-called context-aware applications to trigger specific behaviours. For example, the backlight on a PDA can be adjusted to the current lighting conditions of the environment, as detected by light intensity sensors built into the PDA or embedded in the surroundings. These embedded “context sources” must be discovered by applications or the context management systems supporting the applications, and they must be described using a framework that models their sensing and data processing capabilities, so that appropriate sources can be selected and composed dynamically.

Adopting well-defined and widely accepted standards for sensor communication and description is a preferable approach for achieving interoperability and true plug-n-play. Many standards have been defined over the past decade which tackle this challenge from different directions; however, most of the existing standards failed to reach the desired goal due to a number of functional and/or non-functional reasons. In this paper, we explore a variety of existing standards, which have interoperability as one of the design goals, and adapt these standards into the development of our **Autonomic Context Management System (ACoMS)** that is an open system

where interoperability is absolutely crucial. We argue that the requirements of our context management system can assist the evaluation of these standards from different perspectives of a high-level system. The experiences gained from our development will be valuable not only to the future improvement of our context management system, but also to other sensor-based projects in which standards play a key role in systems integration.

The rest of the paper is organised as follows. Section 2 gives a background introduction to context management in pervasive computing and briefly discusses the challenges that current pervasive computing research is facing, followed by a short introduction of our autonomic context management system. A detailed study of sensor interoperability standards is given in Section 3, followed by our proposed approach to using standards for the development of our system in Section 4. Section 5 discusses our experiences with these standards during the implementation and feasibility study stages of our research. In Section 6, we present our conclusions and future work.

2. CONTEXT MANAGEMENT IN PERVASIVE COMPUTING

In the domain of pervasive computing, “context” is characterised as any information that can be used by context-aware applications to adapt to the current environment or situation. This adaptation is triggered by evaluation of context information gathered from heterogeneous sources, including physical sensing devices, monitoring software, human inputs, and other sources. A context management system, usually viewed as a middleware layer, is responsible for pre-processing and evaluating information gathered from a variety of sources, and disseminating it to multiple context-aware applications. Research shows that model-based high-level abstractions of context information not only removes the burden from resource constrained sources by encouraging context information sharing between many applications, it also allows run-time replacement of sensors with others providing the same sensing phenomena [1], [8]. However, the design of such a context management system poses various challenges, such as (i) **Openness**: future pervasive computing environments will be open and heterogeneous, rather than being constrained to a particular setting with a known set of devices; (ii) **Scalability**: owing not only to the volume of information to be managed

and the dynamic nature of context information, but also to the number of information sources from which context information is gathered; (iii) Interoperability: context sources are as varied as they are numerous in terms of communication protocols, measurement characteristics and data representation; (iv) Flexibility: context information of the same kind can be produced by sources of different sensing types (e.g., location information can be obtained by face recognition systems using conventional surveillance cameras, from location badges, or one of an array of different technologies); and (v) Reliability and Availability: the mobility of both context sources and context-aware applications gives rise to a number of potential issues in the form of disconnections and failures.

Inspired by the vision of autonomic computing, which aims to create self-managing systems in accordance with operation policies and objectives [4], the goal of our autonomic context management system is to provide seamless support for both low-level sensing infrastructures and high-level adaptive applications. This support includes run-time and dynamic discovery of context sources in addition to their configuration and reconfiguration, which mediates low-level complexities inherited from sensing infrastructures, as well as high-level programming abstractions for rapid development of context-aware applications in various domains. Detailed discussion of our approach is outside the scope of this paper and is presented in our previous papers [2], [3].

To address the challenge of heterogeneity of context sources, even partially, we argue that the design of the context management system must be based, to the largest extent possible, on sensor and sensor data processing standards, which are well-defined and widely accepted by industry and the research community. The vision of *invisible computing* [9] can only be achieved when seamless run-time and dynamic sensor discovery, configuration and reconfiguration are possible.

3. OVERVIEW OF SENSOR STANDARDS

The heterogeneity of sensing instrument systems and communication interfaces has been a historical factor contributing to the challenge of creating interoperable sensors for diverse networks, platforms and applications. Same problem also exist in pervasive computing where devices interoperability is needed. For this reason, we studied various open sensor standards which were defined with the goal of addressing these challenges.

Towards addressing heterogeneity of transducers and communication interfaces, the National Institute of Standards and Technology (NIST)¹ has proposed a universal solution, the IEEE 1451 standards family, to offer a standardised way of connecting sensors and actuators to diverse networks and systems. Being a feasible approach for plug-n-play, the IEEE 1451 standards family provides both physical support for the existence of diverse communication interfaces and software solution to model transducer information for autonomic discovery and configuration of transducers.

¹<http://ieee1451.nist.gov/>

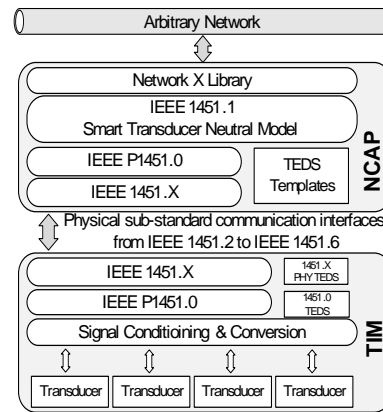


Fig. 1: IEEE 1451 Smart Transducer Standards, where ‘X’ in IEEE 1451.X refers to sub-standards (from IEEE 1451.2 to IEEE 1451.6), which specify different physical transducer communication interfaces (diagram is reproduced from “Smart Sensor Interface Standards IEEE 1451.0 and IEEE 1451.5”, a presentation given in Net-Ready Sensors Workshop by Kang Lee.)

The IEEE 1451 standards family is a collection of “Smart” transducer interface standards that define a set of open, common and network independent communication interfaces for connecting transducers to microprocessors, instrumentation systems and control/field networks. Figure 1 shows a schematic of how each sub-standard fits into the architecture. As you can see from the diagram, two components (in boldface) form the fundamental building block in building network- and vendor-independent interoperable transducers. They are the TIM (Transducer Interface Module) and the NCAP (Network Capable Application Processor). TIM, sometimes referred to as STIM (Smart Transducer Interface Module), is a module providing transducer communication to the higher-level NCAP via the corresponding communication interface specific to each sub-standard, such as UART/TII (IEEE 1451.2), Multi-drop bus (IEEE 1451.3), Mixed-Mode Interface (IEEE 1451.4), a range of wireless network protocols (IEEE 1451.5) and CANopen protocol (IEEE 1451.6). NCAP, first introduced as a core module of IEEE 1451.1-1999², is situated between the smart transducer interface module and an arbitrary network, and provides a communication abstraction for network-independent communications, smart transducer interface module communications, and data conversion functions. A combination of these modules mediates physical and low-level communication with diverse transducers.

Another important feature contributing to sensor interoperability and plug-n-play is the definition of Transducer Electronic Data Sheet (TEDS), which is essentially an electronic replacement of transducer data sheet on paper. The concept of TEDS originated from the IEEE 1451.2-1997³, but later was adopted and redefined by IEEE 1451.4. For

²IEEE 1451.1-1999, Standard for a Smart Transducer Interface for Sensor and Actuators - Network Capable Application Processor (NCAP) Information Model

³IEEE 1451.2-1997, Standard for a Smart Transducer Interface for Sensors and Actuators, - Transducer to Microprocessor communication Protocol and Transducer Electronic Data Sheet (TEDS) Formats

most smart transducers, TEDS is a memory device attached to the transducer, which describes the transducer identification, calibration, correction data, measurement range, manufacturer-related information, and so on.

Although the goal of IEEE 1451 standards family is to provide device interoperability, sub-standards themselves pose a problem in inter-standards communication. In TEDS for example, the sub-standards are defined separately, each with different design goals. The IEEE P1451.0 ('P' stands for preliminary) was, therefore, defined to provide common functions, communication protocols and TEDS formats across the entire standards family.

For the purpose of our research in the pervasive computing domain, we took a closer look into IEEE 1451.4-2004⁴ for the reasons of industry support and refined compact form of TEDS. For a non-functional benefit over the others in the IEEE 1451 standard family, the IEEE 1451.4 is defined and backed by a large number of transducer manufacturers, including National Instruments, Honeywell, Weed Instrument, and others. There are already smart transducer products and development software have been produced and deployed by these smart transducer partners. In IEEE 1451.4, the design of a Mixed-Mode Interface (MMI) allows analog sensing data and TEDS digital data to share the same wire. The refined TEDS description is a bare minimum of the pertinent data stored in a physically small memory device (usually EEPROM) attached to the transducer. For legacy transducers, which lack storage capability, an alternative electronic TEDS (Virtual TEDS)⁵ can be retrieved from a sensor description repository. To further improve memory usage in compact transducers, templates were introduced for a wide range of transducers, such as accelerometers, microphones, strain gauges, thermocouples, thermistors, etc. Each template, written in Template Description Language (TDL), details, bit by bit, the meaning of the TEDS data, and specifies instructions about how TEDS data should be decoded and extracted from its binary encoding. The IEEE standard templates can be concatenated along with manufacturer-customised templates to create a composition of templates for a wide range of transducers that IEEE standard templates alone may not support. The approach of template composition allows transducer specifications defined using IEEE reserved property commands to be obtained by generic parsers, even when manufacturer-customised templates are not available.

The scope of the IEEE 1451 standards family is to provide fundamental support for sensor interoperability. Nevertheless, a limitation of TEDS is that it describes the basic sensor functionality (hardware, sensor calibration, sensing phenomenon, quality of sensor readings) but it cannot capture all the additional descriptions needed by logical/virtual sensors. TEDS is also not able to describe the higher level processing of sensor data (e.g., fusion, interpretation) that sometimes has to be

applied to change the data into the type of context information required by context-aware applications.

These two additional requirements are addressed by the Sensor Web Enablement (SWE) initiative, which is an ongoing activity carried out by the Open Geospatial Consortium (OGC)⁶ with the aim of allowing sensors and sensor data to become discoverable and accessible in real time over the Web. Within the SWE initiative, the enablement of such sensor webs and networks is being pursued through the establishment of a set of encodings for describing sensors and sensor observations, including Observations & Measurements Schema (O&M), Sensor Model Language (SensorML) and Transducer Model Language (TransducerML), and through a number of standard interface definitions for web services, including Sensor Observation Service (SOS), Sensor Planning Service (SPS), Sensor Alert Service (SAS), Sensor Collection Service (SCS) and Web Notification Services (WNS). Among these standard specifications, data pre-processing and fusion is addressed by TransducerML and SensorML.

Essentially, TransducerML⁷ is aimed to standardise a way, in terms of an encoding and a model, for streaming multiplexed data from a sensor system and for describing the system as well as data encoding. In the communication between sensors and TransducerML enabled processors, not only sensor data is transferred, but also the metadata that describes the sensor data. This enables the processor to perform multiple tasks directly on sensor data (searching, discovering, subscribing, and processing), rather than purely analysing or acting on sensor data. In order to identify snapshots of sensor data, sensor data is time tagged according to a common clock.

SensorML is defined primarily for describing sensor systems and observation processing. With the capability of its standard model and XML schema, SensorML provides a flexible and extensible description framework to share information needed for discovering sensors, geolocating sensor observations, processing low-level sensor observations, and exploring a variety of properties associated with current tasks. The core of the SensorML model is the modelling of all components as processes. These components include individual sensing devices, complex sensor systems and data processing modules. Each individual process defines its inputs, its outputs and methods for processing its inputs and parameters to yield its outputs. Processes can be composed to create process chains, which can be further composed in a *filter-and-pipe* [7] fashion. SensorML also provides comprehensive support for describing a variety of sensor related specification information.

Standards described above can be combined to minimise the use of memory for storing and processing sensor specifications at the hardware level and to provide dynamic discovery of sensors and sensor capabilities at the application level. One possible combination of these standards is to use TEDS of IEEE 1451 for enabling sensors to describe themselves, and then to use SensorML to model these descriptions in an extensible and

⁴IEEE 1451.4-2004, Standard for A Smart Transducer Interface for Sensors and Actuators - Mixed-Mode Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats

⁵<http://www.ni.com/teds>

⁶<http://www.opengeospatial.org>

⁷<http://www.transducerml.org>

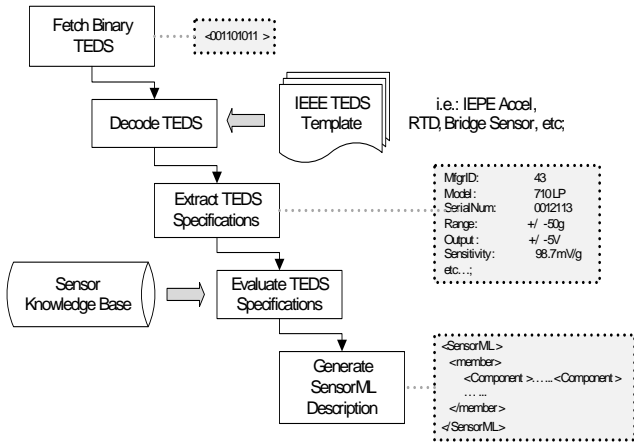


Fig. 2: Process of Mapping TEDS to SensorML; snapshot example data in each step is presented in dotted grey boxes.

flexible representation for higher-level systems/applications, while O&M and TransducerML are exploited to describe the properties of observations (measurements) gathered from sensors, and to encode streaming multiplexed data from sensor systems respectively. In the next section, we present a method for translating TEDS descriptions into SensorML.

4. MAPPING TEDS INTO SENSORML

One of our ultimate goal, as discussed in Section 2, is to provide transparent context source management support for context-aware applications. With the knowledge we gained from a study of sensor standards in section 3, we present our approach in mapping IEEE 1451 (IEEE 1451.4-2004 for the reasons described earlier) TEDS into SensorML to provide “post discovery” management of context sources. Figure 2 illustrates the steps and the components involved in this mapping process. Hardware level support for interoperability has been briefly discussed; a more detailed discussion of this topic is outside the scope of this paper.

As mentioned in Section 3, transducer communications are handled by NCAP and corresponding TIM of the sub-standard (the MMI in the case of IEEE 1451.4-2004). Transducer TEDS is accessible via the MMI in binary format. The retrieved binary TEDS data is then decoded using an appropriate transducer template. The TEDS data comprises of a block of basic TEDS (including manufacturer ID, model number, serial number, version number and version letter), template TEDS and user data. A transducer template written in TDL serves as a ‘map’ to instruct a decoder as to how each property field should be read. A list of reserved property commands are defined and maintained by IEEE for parsing transducer specifications, examples of these property commands are shown in Table 1. Manufacturers of smart transducers are also able to extend the list with customised property commands to meet their needs.

The mapping between TEDS specifications and SensorML descriptions is done by creating a knowledge base that maps each TEDS property command, which may have a number

TABLE 1: EXAMPLES OF MAPPING, EACH SENSORML DESCRIPTION IS DESCRIBED AS IN FIGURE 3

TEDS Commands	Description	SensorML Description
%System_ModelNum	Model Number	ModelNumber
%ElecSigType	Electronic Signal Type	SensorType
%SensorImped	Sensor Impedance	SensorImpedance
%MaxPhysVal	Max Physical Value	MaxPhysicalValue
%User	User Information	UserData

```

<smil:identifier name="ModelNumber">
  <smil:Term definition="com:nicta-x:def:identifier:ModelNumber">
    <smil:value>NICTA-123456</smil:value>
  </smil:Term>
</smil:identifier>

```

Fig. 3: A generated example of SensorML Identification Description

of synonyms, to an appropriate SensorML description. As illustrated in Figure 2, a sensor knowledge base is created to capture expert knowledge in interpreting each transducer property. These knowledge bases are usually defined in the form of dictionaries, registries or ontologies. Table 1 shows some examples of these mappings. For instance, the TEDS specification (%System_ModelNum, NICTA-123456) can be mapped into a SensorML description as *Identification*, which is referring to a unique ontology resource, as shown in Figure 3. As a last step in creating a sensor description for higher-level application/system requirements, a SensorML description will be generated.

The prototype implementation consists of two parts: a TEDS decoding/extraction component, which is able to decode binary TEDS and extract specifications according to a sensor template written in TDL; and a SensorML engine, which generates SensorML descriptions (based on the latest approved version V1.0 of the SensorML standard specification, OGC Specification 07-000) according to the supplied TEDS specifications .

In addition to the mapping of TEDS to SensorML, we also conducted a small feasibility study on functionalities of the IEEE 1451 standards family using our wireless sensor hardware, the Crossbow Mote⁸. The idea of this feasibility study is to see how a non-IEEE 1451 compliant sensor product can benefit from these standards with minimal extra work. In this study, we modified the structure of TEDS and redefined a customised template using TDL, which reflects the physical specifications of the temperature sensor, a Panasonic NTC thermistor⁹ mounted on the MTS 310 sensor board. The property commands list has also been extended based on the IEEE list for smart transducers. For example, a privacy and security measure is added to the customised TEDS data in order to facilitate privileges for sensor control. TEDS data is encoded using the aforementioned customised template and stored into the configuration block of TinyOS¹⁰ which is the operating system running on our Mote hardware. A similar modification of IEEE 1451 is suggested in [6], but they are more focused

⁸<http://www.xbow.com>

⁹<http://www.panasonic.com>

¹⁰<http://www.tinyos.net>

on the modification of low-level components of the IEEE 1451 standard, such as IEEE 1451.1 modules. The study shows that it is feasible to enhance existing wireless sensor nodes (i.e.: Mote family platforms) with self-describing capability. However, the study also suggested some improvements that might be made to the IEEE 1451 standard. These suggestions are provided at the end of Section 5.

5. LESSONS LEARNED AND SUGGESTIONS

The potential for adopting open sensor standards to enable sensor interoperability and plug-n-play is appealing. The IEEE 1451 standards family mediates the diversity of underlying communication interfaces of transducers and provides a network-independent communication abstraction for arbitrary networks, while SensorML allows modelling of sensor information in an expressive and extensible description framework. But there are, however, some limitations in the standards and in the process of harmonising the standards. In this section, we present a summary discussion of our experiences in adapting these sensor standards with the goal of developing our autonomic context management system. However, we believe these experiences should apply to other open sensor based systems challenged by similar interoperability issues, because the digitisation and standardisation of sensor specifications and their self-describing functionality enables dynamic discovery and re/configuration of sensors. These features also maximise the capability of data processing and eliminates man-made errors in interpreting sensor specifications.

The immediate concern with regard to the limitations is the consistency of standards in terms of maintenance and management. It is a non-functional consideration, but crucial to determining whether a standard like IEEE 1451 can be more widely accepted and adopted by manufacturers. We question whether the standards specifications and related elements (transducer templates and property commands in the case of IEEE 1451.4) can be appropriately managed in a way that allows new smart transducers available in the market to be supported. The list of manufacturer IDs and transducer property commands are maintained by IEEE, which means manufacturers of smart transducers must register a unique manufacturer ID from IEEE and make available all customised templates and property commands in order to ensure their products are IEEE 1451.4 compliant. This procedure normally involves time-consuming reviewing processes and a waiting period. It is one of the reasons why many existing standards have failed to gain wide acceptance and many standards compliant devices are not truly plug-n-play. To resolve this issue, the development of applications/systems should provide a 'backward' compatible solution for generic functionalities of transducers in the case that the innovative features are not supported. To implement such 'backward' compatibility, our implementation will only map TEDS data of known property commands; other unknown properties are mapped into an *Identification* element in SensorML description for human interpretation.

A fundamental step towards harmonising IEEE 1451 and SensorML standards is to define mappings of terminologies between standards. These mappings must be carefully defined, ideally by domain experts, to fulfil most of the needs for modelling a wide range of smart transducer specifications. For the purpose of studying with the capabilities of IEEE 1451 and SensorML, we defined direct mappings from each TEDS property command to a SensorML description, which then refers to a unique ontology resource. In our approach, mappings and ontologies are exploited to capture high-level expert knowledge in interpreting specific sensor characteristics and capabilities. It is a common model for capturing human knowledge in various computer science areas, such as expert systems. With the same vision of harmonising a variety of sensor standards to provide better interoperability support for other sensor based projects, the Semantic Interoperability Community of Practice¹¹ (SICoP) has formed a working group with the aim of defining a knowledge base, called SSHWG, to harmonise a number of standards. At the moment, standards like IEEE 1451, ANSI N42.42, etc. are included in the SSHWG, and a plan to include SensorML and TransducerML is underway. Concurrently, the standard organisations NIST and OGC also highlighted the importance of their standards to be interoperable and claimed that they plan to enable bidirectional interoperability; however, there is little information available about their implementation plan. Nevertheless, one direct concern is that a complete solution for harmonising these two standards will introduce additional complexities, which might eventually contravene their original simplicity when used in isolation. There is currently no consensus as to whether the best way forward is to harmonise these standards or to introduce a new standard that combines the capabilities of the two.

The final consumer of the generated SensorML description is a variety of higher-level applications/systems; in our case, it is the autonomic context management system that exploits these SensorML descriptions to perform transparent management of context sources for multiple context-aware applications. In order for these consumers to better understand the meaning of the terminologies used in the description, a semantic definition for each term is necessary. However, OGC only recommended a limited number of terminology definitions; this limits the expressiveness of SensorML for a wide range of sensors. To bridge the gap between IEEE 1451 and SensorML, in our experiment, we created an ontology for terminologies according to the list of TEDS template property commands. The appropriate form of the knowledge base is a matter of contention; however, the use of an ontology to model terminologies allows each term to be identified with its unique namespace in order to avoid name conflict when terms from multiple domains are merged together. Building a general ontology for cross-domain knowledge is a very important but difficult challenge as it requires collaboration between each knowledge domain. Separate works are being carried

¹¹<http://colab.cim3.net/file/work/SICoP/2007-02-27/>

out by the Marine Metadata Interoperability community¹² and the Semantic Web for Earth and Environmental Terminology (SWEET)¹³ to achieve interoperability of terms in their specific domains; however, at this stage they are incomplete. The completeness of ontologies for term definitions is an important evaluation factor to the success of SensorML modelling.

We realized, from our experiment, the ideal memory size of 256 bits for TEDS data, as suggested by the standard, is difficult to achieve. To relax the memory problem, the author of IEEE 1451.4 suggests manufacturers increase the memory size accordingly. However, it does not solve the underlying issue for memory-constrained devices. During our feasibility study when attempting to encode the target temperature sensor using the IEEE standard template for thermistors (Template ID 38), we realized the encoding is strictly tied to the order of each property command described in the template. This is due to the 'lookup-based' template encoding/decoding mechanism where the position and size of a property command in the template indicate the order and number of bits that should be read from the TEDS binary. Furthermore, the number of property commands required to describe a transducer varies greatly. For example, to model the thermistor in our study, three property commands (the Steinhart-Hart Coefficients: %SteinhartA, %SteinhartB and %SteinhartC) are not specified by the actual hardware, and therefore are not needed, but each takes 32 bits (96 bits in total). However, there is no feature provided to overcome this problem in the IEEE 1451 standard, except redefining the sub-template to avoid them in the encode/decode procedure.

In order to enhance memory utilisation of the template encoding mechanism, we propose a modified TEDS encoding format, which uses an array of flags for each template to indicate all required property commands when describing a transducer. Each bit in this array indicates whether a property command is *enabled* or *disabled*, where *enabled* means the current property command is required and the decoder should read the appropriate number of bits from the TEDS binary and *disabled* means skipping the current property command from the binary. The order of bit flags is determined by the order of property commands in the template it represents. For every template, an array will be placed just before the template encoded TEDS binary. We evaluated the efficacy of our approach against the existing template encode/decode mechanism, and we realised our proposed TEDS format not only allows TEDS data to be encoded/decoded in a flexible and memory efficient way with no modification required to the template encoding concept, but also eliminates the need to define separate customised templates for each transducer model where the number of property commands required vary.

In addition to refining the TEDS binary encoding, we also suggest that, based on our experiences, the IEEE standard should also consider providing descriptive information for each Transducer Interface Model (TIM) in the same way as

one TEDS description for every transducer, since each TIM may offer different communication interfaces to communicate with higher-level systems/applications, and/or may pre-process (aggregation, averaging, etc) low level sensing data before forwarding to consumer systems. We argue that this information should be described and made available along with the transducer TEDS for automatic discovery and configuration of TIM.

6. CONCLUSION AND FUTURE WORK

In this paper, we described our experiences of applying sensor standards to our autonomic context management system and conducting a small experiment. Standards based approaches offer a range of advantages over proprietary solutions; however, both functional and non-functional concerns of standards place limitations on the practical application of these standards in other research areas. These limitations include consistency of standards maintenance and management, and completeness of ontologies for term definitions.

The current implementation of our system does not include TransducerML encoding support for streaming data. We plan to incorporate this encoding in order to support multiplexed sensor data. As ontologies for term definitions determine the usability of SensorML description models for higher-level applications/systems, we will extend our ontologies to widen the domain of knowledge, ideally eliciting this knowledge from domain experts.

7. ACKNOWLEDGEMENTS

NICTA 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.

REFERENCES

- [1] K. Henricksen and J. Indulska. A software engineering framework for context-aware pervasive computing. In *Pervasive Computing and Communications, 2004. PerCom 2004. Proceedings of the Second IEEE Annual Conference on*, pages 77–86, 2004.
- [2] Peizhao Hu, Jadwiga Indulska, and Ricky Robinson. Reconfigurable middleware for sensor based applications. In *MDS '06: Proceedings of the 3rd international Middleware doctoral symposium*, page 5, Melbourne, Australia, November 2006. ACM Press.
- [3] Jadwiga Indulska, Karen Henricksen, and Peizhao Hu. Towards a standards-based autonomic context management system. In *3rd International Conference on Autonomic and Trusted Computing (ATC) '06*, 2006.
- [4] Jeffrey O. Kephart and David M. Chess. The vision of autonomic computing. *Computer*, pages 41–50, 2003.
- [5] Belle Mellor. A world of connections. *The Economist*, 2007.
- [6] R.L. Oostdyk, C.T. Mata, and J.M. Perotti. A kennedy space center implementation of ieee 1451 networked smart sensors and lessons learned. In *Aerospace Conference, IEEE*, page 20pp., 4-11 March 2006.
- [7] M. Shaw and D. Garlan. *Software Architecture: Perspectives on an Emerging Discipline*. Prentice Hall, 1996.
- [8] T. Strang and C. Linnhoff-Popien. A context modelling survey. In *First International Workshop on Advanced Context Modelling, Reasoning And Management UbiComp 2004*, Nottingham, England, 2004.
- [9] Mark Weiser. The computer for the twenty-first century. *Scientific American*, pages 94–10, September 1991.

¹²<http://marinemetadata.org>

¹³<http://sweet.jpl.nasa.gov/ontology/>