A Comprehensive Ontology for Knowledge Representation in the Internet of Things

Wei Wang, Suparna De
Centre for Communication Systems Research,
University of Surrey, Guildford
United Kingdom
e-mail: {Wei.Wang, S.De}@surrey.ac.uk

Ralf Toenjes, Eike Reetz
Faculty of Engineering and Computer Science
University of Applied Sciences
Osnabrück, Germany
e-mail: {r.toenjes, E.Reetz}@hs-osnabrueck.de

Klaus Moessner
Centre for Communication Systems Research,
University of Surrey, Guildford
United Kingdom
e-mail: {K.Moessner}@surrey.ac.uk

Abstract—**Semantic modeling for the Internet of Things has become fundamental to resolve the problem of interoperability given the distributed and heterogeneous nature of the "Things". Most of the current research has primarily focused on devices and resources modeling while paid less attention on access and utilisation of the information generated by the things. The idea that things are able to expose standard service interfaces coincides with the service oriented computing and more importantly, represents a scalable means for business services and applications that need context awareness and intelligence to access and consume the physical world information. We present the design of a comprehensive description ontology for knowledge representation in the domain of Internet of Things and discuss how it can be used to support tasks such as service discovery, testing and dynamic composition.**

Keywords-Internet of Things, Semantic Modelling, IoT Services, Service Testing, Ontology

I. INTRODUCTION

Advancement in wireless sensor networks has led to a potential interest in integrating data and capabilities provided by physical world objects into the Internet. The Internet of Things (IoT) refers to interconnection of the objects or things in the physical world and their virtual representations on the Internet. As one of the fundamental constituents of future Internet, IoT has attracted tremendous interests from various research communities and industry. During the past few years, scope of the research and development has been extended substantially, from the original focus on things traceability and accessibility using RFID tags to IoT infrastructure and architecture, communication protocols for constrained devices, (mobile) sensors and sensors networks, smart things, middleware, security and privacy, and many others. Among these developments semantic oriented computing manifests its potential to cope with the challenging problems of heterogeneity and interoperability exposed by the large number of things with different characteristics.

Already, we have seen many applications using semantic Web technologies in IoT research, in particular the SSN ontology [1] for annotating sensors and sensor networks; Linked Data [2] for sensor data publishing [3] and discovery [4], and semantic sensor observation services (SemSoS) [5]. The recent work in [6] and [7] proposes a modeling approach in which resources on the IoT are able to expose standard service interfaces. This coincides with the principle of the Service Oriented Computing [8] and potentially provides a scalable, distributed and service-oriented means to access IoT information. More importantly, existing methods for service discovery and composition can easily access IoT based services to create context-aware and personalised services and applications.

Semantic modeling for the IoT domain provides a basis for interoperating among different systems and applications; however, current work has mostly focused on IoT resources management while not on how to access and utilise information generated in IoT. In this paper, we present a description ontology for the IoT domain by integrating and extending existing work in modeling concepts on the IoT. The ontology helps exploit the synergy of the existing efforts and provides support for crucial tasks in such as IoT resource and service discovery, IoT service testing, composition, adaptation and etc. The ontology is compatible with several widely used semantic models in IoT and is designed to be lightweight to promote reuse and support more efficient inference. The rest of the paper is organised as follows. In Section 2, we review some of the representative semantic modeling methods for domain of IoT. Section 3 presents an overview of a lightweight description ontology for the domain of IoT and its relations to existing models. In Section 4, we focus on discussion of three important modules in the description ontology, i.e., IoT service modeling, Quality of Services (QoS) and Quality of Information (QoI), and IoT service test. In section 5, we discuss some of the issues and challenges related to applying the description ontology in service testing, composition and adaptation. We conclude the paper and briefly discuss our future work in Section 5.

II. RELATED WORK

The life cycle and development of IoT services should consider testing and validity of services based on the multiplicity of application contexts and dynamic environment changes in IoT platforms. In this section we present the state-of-the-art on some of the key themes such as IoT resource and service modeling, and service testing technologies.

The IoT-A project has identified entities, resources and IoT services as key concepts within the IoT domain [9]. The entity is the main focus of interactions by humans and/or software agents. IoT services expose resource functionality hosted on devices that provide some forms of physical access to the entity. An entity is modeled to have attributes that tie it to the domain (i.e. observable or actionable features), location attributes as well as type and identifier specifications. Also captured are optional temporal features and links to known vocabularies for specifying ownership. The resource model captures different resource types (e.g. sensor, actuator, RFID tag), hosting device location as well as a link to the service model that exposes the resource capabilities. The service model exposes resource functionalities in terms of the IOPE (input, output, precondition, effect) aspects. The type of the service specifies the actual technology used to invoke the service (e.g. OWL-S, REST etc.). Two important aspects captured in the service model are the service area and the service schedule. For sensing services, the service area would be the observed area, while actuating services would specify the area of operation. The possibility of specifying time constraints on service availability is captured through the service schedule feature. The IoT-A models form the basis of the model proposed in this paper, which are extended to encompass service quality and testing aspects.

Standardisation efforts in the allied areas of sensor description and observation data modeling have been driven by the W3C's Incubator Group on Semantic Sensor Networks (SSN) [10] and the OGC Sensor Web Enablement [11] suite of XML-based standards. The SSN ontology [1] represents a high-level schema model to describe sensor devices, their capabilities, platform and other related attributes in the semantic sensor networks and the sensor Web applications. The SSN ontology, however, does not include modeling aspects for features of interest, units of measurement and domain knowledge that need to be associated with the sensor data to support autonomous data communications, efficient reasoning and decision making. The OGC standards suite is aimed at Web accessible sensor networks and archived sensor data that can be discovered and accessed using standard protocols and APIs. The standards consist of modeling schemas (Observation and Measurement (O&M) and SensorML [12]) and Web Service interfaces (Sensor Alert Service, Sensor Planning Service and Sensor Observation Service) that facilitate the exchange of information through APIs. The research work by Henson *et al.* [5] provides a semantically enabled Sensor Observation Service, called SemSOS, which provides the ability to query high-level knowledge of the environment as well as low-level raw sensor data. 52North's [13] SOS implementation is designed to provide a Servlet interface to sensor observation data stored in PostGIS database, with the sensor descriptions stored in XML files. The work presented in [14] proposes an ontology-based model for service oriented sensor data and networks. The ontology consists of three main components: ServiceProperty, LocationProperty, and PhysicalProperty. ServiceProperty explains the functionality of a service, while properties in the other two components describe contextual and physical characteristics of the sensor nodes in wireless sensor network architecture. The system, however, does not specify how sensor data will be described and interpreted in a sensor network application.

Ontology-based testing techniques have been applied to Web Services as well as multi-agent systems. The main focus of these approaches is to automate the test case production process and input data generation. The approach presented in [15] uses interaction ontologies that define the semantics of agent interactions to automatically generate test cases. Instances of the interaction ontology are augmented with domain ontology instances and form the input test data. With the captured semantics, inputs are generated based on ontology property assertions and constraints on input values. A test ontology model has been proposed in [16] for generating test cases for OWL-S Web Services. The approach generates data pools by analysing the IOPE parameters of the service specification and then creating data partitions for each pool by deriving the restrictions and relationships of the partitions. However, for test cases to be generated automatically by service creation environments, test functionalities need to be built into the service lifecycle at design time in order for the test framework to be able to verify domain parameters and variables.

III. OVERVIEW OF THE DESCRIPTION ONTOLOGY

The description ontology is developed using a knowledge-driven approach and aims to capture most of the important concepts and their relationships in the IoT domain. The linked data principle is fundamental for the design: concepts that are isolated in existing works are linked to each other as well as to external domain ontologies and even the open linked data cloud.

A. Design Principles

The major consideration in our design is to balance the tradeoff of being lightweight and complete. The ontology is designed based on the following four principles:

(1) Lightweight: experiences on ontology development in the past years show that a lightweight ontology model that well balances expressiveness and inference complexity is more likely to be widely adopted and reused.

(2) Completeness: we aim to develop a more complete description ontology for the IoT domain by integrating and extending existing works on IoT modeling. Users of the ontology can exploit the synergy of integration to support common tasks in IoT.

(3) Compatibility: the ontology needs to be consistent with those well designed, existing ontologies to ensure compatibility.

(4) Modularity: the designed ontology is developed with a highly modular approach to facilitate its evolution, extension and integration with external ontologies.

B. Ontology Modules

The description ontology contains seven main modules, namely, IoT Services, Service Test, QoS and QoI, Deployment, System and Platform, Observation and Measurement, IoT Resources, and Entity of Interest and Physical Locations. Figure 1 shows an overview of the description ontology. For some of the modules in the description ontology only properties are defined; this allows users to link to the concepts in external ontologies or existing linked data.

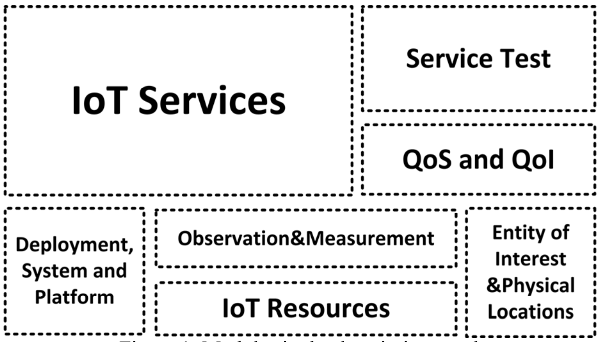

Figure 1. Modules in the description ontology

(1) IoT resources: existing works for modeling the IoT resources primarily focus on sensors and sensor network [3], [4]. This module extends the SSN ontology [10] by including other important resources in the IoT domain such as Actuator, IoT Gateway and Server.

(2) IoT services: the scale and distributed nature of the IoT requires scalable and interoperable means for managing and accessing information pertaining to the physical world. With the service interfaces exposed by the IoT resources, existing business applications and services need intelligence and context awareness could be easily integrated with the low level IoT services. An important consideration is to model IoT services in a way that adheres to existing service standards (e.g., SOAP and REST based services).

(3) Quality of Services (QoS) and Quality of Information (QoI): QoS and QoI have been important concepts in many areas such as networking, communication and Web services. IoT features a vast number of energy-constrained and mobile resources with limited computation power that usually operate in harsh and dynamic environments. This makes QoS and QoI particularly important in service composition and adaptation for IoT service providers and consumers.

(4) Service Test: the test components are proposed for testing and verifying functional and non-functional capabilities of IoT services during design and deployment stages. They are aligned with the services components by linking them to the concepts in the process ontology of OWL-S.

(5) Deployment, Systems and Platforms: This module provides descriptions on how the IoT resources are organised and deployed as well as the system they form. Modeling and linking together these concepts enable a high-level view on relationships among the IoT resources and the systems and platforms that support them.

(6) Observation and Measurement: concepts in this module represent the information collected from the physical world by the IoT resources. We reuse the concepts related to Observation and Measurement from the SSN ontology.

(7) Entity of Interest and Physical Locations: Entity of Interest represents an object in the physical world that is of interest to a user or application. Physical locations are associated with entity of interest and essential for IoT resource and service discovery.

Modeling methods on the IoT Resources, Entity of Interest and Physical Locations, Deployment, Systems and Platforms, and Observation and Measurement have been extensively discussed in existing works such as [1], [6], [4]. We extend these works with a particular emphasis on those modules that facilitate us to access, utilise and verify information generated by the IoT resources. In the next section, we present our modeling approach for IoT services, QoS and QoI, and Service Test.

IV. MODELING OF IOT SERVICE AND TESTING

IoT services are exposed by IoT resources and mostly provide (near) real-time and transient information on the physical world through standard service interfaces. They usually operate in harsh and dynamic environments where the resources (e.g. battery, computing and communication capabilities) are constrained and may appear or vanish suddenly; therefore, IoT services have to include testing from the beginning. It is expected that a large number of IoT devices will demand for self-testing capabilities. The dynamic environment also brings significant needs for service adaptation and even re-composition. In this section, we focus on describing concepts related to IoT services and service test that can be used to address the identified challenges.

A. IoT Services

Compared to the well-engineered high-level business services, IoT services tend to be less reliable because of the nature of IoT resources and their operating environments. This necessitates the need for testing throughout the service lifecycle and additional mechanisms for service adaptation and re-composition (e.g., QoS and QoI of IoT services). We will discuss these mechanisms in the following sections.

We define IoT service as a subclass of the Service class defined in the OWL-S [18], therefore, an IoT Service can have one Service Profile and one Process that describe its functional and non-functional properties (these are inherited from the OWL-S Service class). A relationship “exposes” (and its reverse property “isExposedBy”) is defined between An IoT Resource and an IoT Service as shown in Figure **2** (due to the space constrains, only part of the ontology is shown in this paper. The complete ontology and a description of a temperature sensor service are available at:

http://ccsriottb3.ee.surrey.ac.uk:8080/IotaDataFiles/models/IoT.est.owl and http://ccsriottb3.ee.surrey.ac.uk:8080/IotaDataFiles/models/iotestExample.owl, respectively). An IoT service can link to one or more instances of the test class through the property "hasTest" (see Figure 4). We also define different types of Test such as functional test, reliability test, unit test, etc, all of which can be defined based on the users' needs. The QoS and QoI concepts are defined as two different classes and linked to the IoT Service class through the "hasQoS" and "hasQoI" relations. Values of the different parameters under QoS and QoI would be constantly changing due to the changes in the environments, so they can be used as essential criteria for performing service adaptation and even re-composition in pervasive environments.

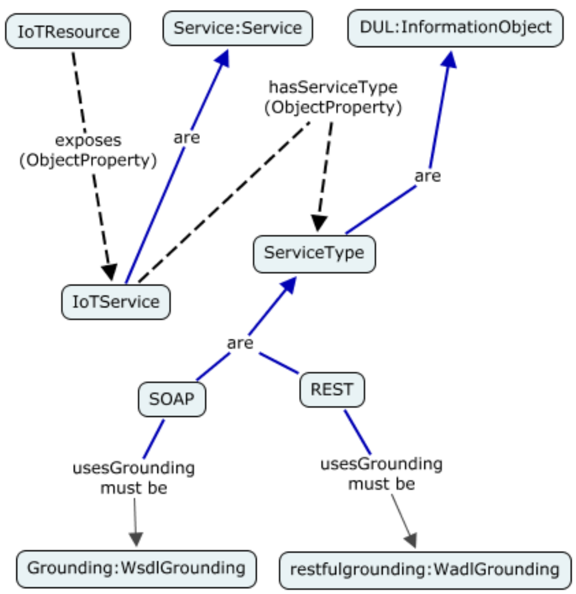

Figure 2. IoT Service and Resources of the description ontology

Another important design consideration is which service technologies to use for IoT services. The SOAP based services (described using WSDL) have strong associations with business process modeling and have been widely adopted in the business world, while REST style services are data-centric and have been prevalent in Web 2.0 applications recently due to their flexibility and simplicity [17]. RESTful services are usually described using the Web Application Description Language (WADL). In our design, we assume that IoT services will be designed using either SOAP or REST based methods and we use OWL-S as the primary language for IoT service description. Profile, Process and Service ontologies in the OWL-S [18] are used to describe the functionalities and processes for IoT services. For services grounding we use the class WSDLGrounding for SOAP based ones and WADLGrounding (which is a subclass of the OWL-S Grounding [17]) for REST based ones. This ensures design consistency through the use of a unique language for describing different types of services technologies (only the Groundings differ) and maintains the service property of being process-oriented, i.e., RESTful services can also participate service composition with SOAP based services.

B. Quality of Service and Quality of Information

QoS and QoI have been extensively studied in many areas such as networking and communication [19], Web services [20], and can be used as important criteria for designing complex service composition and adaptation algorithms [21]. They are particularly important for the IoT domain which exhibits a much higher level of dynamicity. In our work we do not try to enumerate and model all the parameters for QoS and QoI since they are often application dependent. Instead we define the parameters that are common to many application domains. In the current version of the ontology, both QoS and QoI are modeled as classes (with a number of subclasses for each) and linked to both IoT Service class and IoT Resource class. QualityOfService is defined as the top-level QoS class that has networking related subclasses (e.g., Throughput and Delay), Availability, Reliability, Security, etc; QualityOfInformation has subclasses such as Correctness, Precision, Provenance, etc, as shown in Figure 3. All these classes have the properties of "CalculationValue" (value of the QoS or QoI paramter) and "CalculationMethod" (method for calculating the QoS or QoI value). The range of the "CalculationMethod" property is a computation method that can be represented using appropriate expressions or URIs to facilitate the reuse of QoS or QoI information.

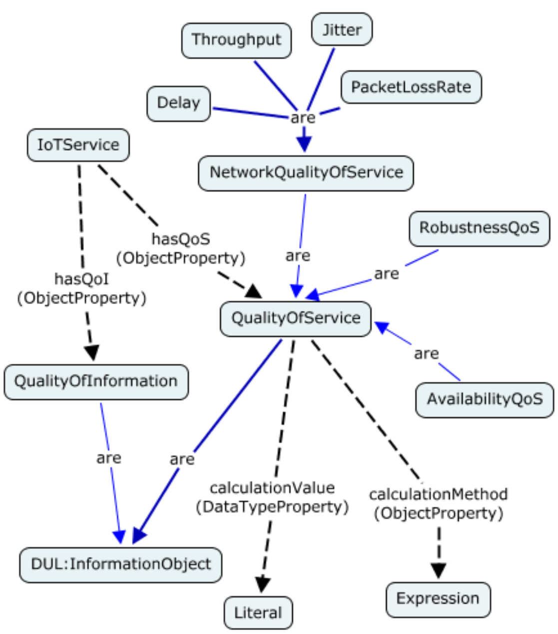

Figure 3. QoS and QoI in the description ontology

C. Test for IoT Services

The distributed and heterogeneous nature of IoT Services demand for strong Test capabilities: for example, to ensure the correct functionalities of the services during design and deployment phases; to ensure that the performance of the services meets the users' requirements as well as service level agreements between service providers and consumers. The procedure for testing is closely related to the service process modeling and the service models described above are used to describe test cases, test data and the test flow. This work employs the concepts defined in the OWL-S [18] to specify the service test semantics, including inputs, outputs, preconditions and effects (named IOPE).

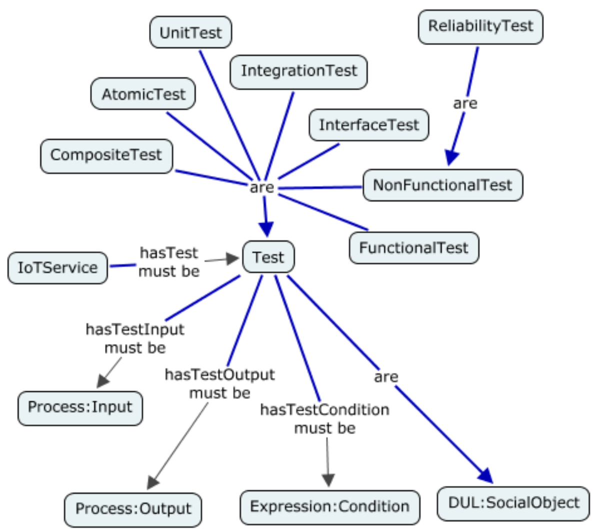

Figure 4 Test Description Ontology

The description of the test model within a test ontology is the knowledge base for automated test cased creation and execution. The Test Description Ontology enables the creation of re-usable test cases and the modeling of the test executions flow. Our description ontology defines atomic tests and composite tests, distinguishing different types, e.g. functional or reliability, and levels of tests, i.e. unit, interface, integration, or collaboration test. Service attribute values are constrained by a Min and Max Value or a Value List and an optional Default Value. Moreover, Timers are defined to describe transition of states and events. . Figure 4 shows a part of the test component of the description ontology. Description of the service test facilitates automated test generation for the service under test. Reasoning engines, e.g. rule based systems, can exploit the knowledge to derive the behaviour model and constrain the test cases. The test cases are organised in test suites and a test plan defines the scheduling of test execution, e.g. sequential, concurrent and process forking.

V. DISCUSSION

IoT services represent a fundamental class of services in the service domain, enabling high-level business services and applications to incorporate context awareness and speculate on personalisation. Description of the IoT service needs to support efficient service discovery and composition; however, given the tremendously large number of IoT resources this is not a trivial task. Besides describing the functional properties of the services (e.g., input, output, precondition and effect), the description ontology also provides descriptions in terms of non-functional properties (e.g., QoS and QoI) as well as links to domain knowledge (e.g., service category and physical location ontologies).

QoS and QoI become more essential because high level business services and applications which utilise the IoT data need to monitor them in order to choose more reliable and quality IoT services. Moreover, they serve as important criteria in decision-making for service adaptation and re-composition. The fact that values of the QoS and QoI parameters are constantly changing along with the changes of the physical environment necessitates effective monitoring mechanisms for IoT services. This might become impractical or inefficient when there are extraordinarily large number of IoT service instances. Instead of monitoring the IoT services, more scalable methods such as event reporting and complex event processing techniques can be used to derive events and update the values of QoS and QoI parameters.

Current solutions for IoT service creation neglect the need for automated mechanisms for test generation. The UML 2.0 Testing Profile (U2TP) [22] provides methods for designing and specifying tests but lacks tools for test execution. Alternatively, the Testing and Test Control Notation Version 3 (TTCN-3) [23] provides means to specify tests and control the execution. However, the test cases are usually defined manually by an experienced tester. Both test languages miss semantics to guide automatic generation of tests.

We propose a model based approach to guide automatic test generation and control. The model provides a partial description of the service under test (SUT) (see Figure 5). The partial description has to cover the test relevant aspects, e.g. functionality of the SUT. The service behaviour can be efficiently modeled by finite state machines [24]. Subsequently a path search algorithm can analyse the behaviour model and derive from each path through the FSM a possible test case, defining together with the test data the abstract tests, specified in TTCN-3. Finally the TTCN-3 test cases are compiled and run on the test environment within the TTCN-3 test framework.

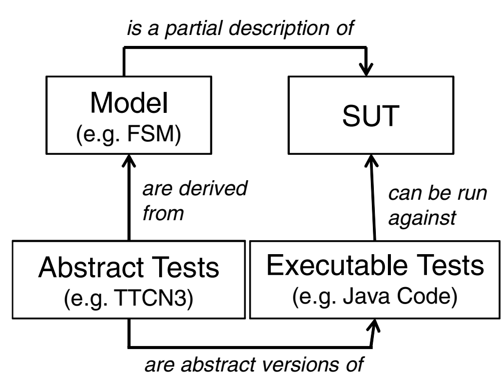

Figure 5. Model based Testing

The automated generation of executable tests from abstract tests has been shown in [24]. However, the automatic test case generation is currently hindered particularly in the two other steps: i) the generation of the behaviour model and ii) the efficient definition of test cases and test data ensuring reasonable test coverage while avoiding an exaggerated number of tests. This is where the knowledge base represented by OWL-S is employed to guide test automation.

VI. CONCLUSION AND FUTURE WORK

Modeling using semantic Web technologies has shown effectiveness for supporting interoperability among large number of resources on the IoT in many existing works. Recently, the research trend has shifted from IoT resources to IoT information, since the ultimate goal of the IoT research is to enable ubiquitous access and utilisation of the physical world information, especially for high level business services and applications that need context awareness and intelligent decision making. An interesting idea in this line is to provide IoT information through standard service interfaces, which coincides with the service oriented computing paradigm and ensures scalability. To this end, a description ontology that balances the tradeoff between being comprehensive and lightweight is needed to capture and represent knowledge for the IoT domain, and to support the common tasks such as resource management and discovery, service composition, adaptation and testing. The description ontology we present here integrates the existing efforts for modeling the IoT domain concepts and is extended with essential concepts such Testing to ensure correct functionality of IoT services at both design and runtime stages. It also contains QoS and QoI modelling which is particularly important for IoT based service composition and adaptation. The description ontology presented in this paper is developed in an ongoing IoT research project. Our future work involves development of efficient and QoS and QoI aware methods for service composition and adaptation in dynamic environment.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Commission’s Seventh Framework Programme for the IoT.est project under grant agreement no. 288385.

REFERENCES

[1] W3C. (2011, W3C SSN Incubator Group Report. Available: http://www.w3.org/2005/Incubator/ssn/wiki/Incubator_Report

[2] C. Bizer, T. Heath, and T. Berners-Lee, "Linked Data - The Story So Far," International Journal on Semantic Web and Information Systems (IJSWIS), 2009.

[3] P. Barnaghi, M. Presser, and K. Moessner, "Publishing Linked Sensor Data," in Proc. 3rd International Workshop on Semantic Sensor Networks (SSN), in conjunction with the 9th International Semantic Web Conference (ISWC 2010), 2010.

[4] J. Pschorr, C. Henson, H. Patni, and A. Sheth, "Sensor discovery on linked data," in Proceedings of the 7th Extended Semantic Web Conference, ESWC2010, Heraklion, Greece, 2010.

[5] C. Henson, J. K. Pschorr, A. P. Sheth, and K. Thirunarayan, "SemSOS: Semantic Sensor Observation Service," in Proc. of the 2009 International Symposium on Collaborative Technologies and Systems (CTS 2009), Baltimore, MD, 2009.

[6] S. De, P. Barnaghi, M. Bauer, and S. Meissner, "Service modelling for the Internet of Things," in Proceedings of the Federated Conference on Computer Science and Information Systems, Szczecin, Poland, 2011, pp. 959-965.

[7] A. Serbanati, C. M. Medaglia, and U. B. Ceipidor, "Building blocks of the internet of things: State of the art and beyond," in Deploying RFID - Challenges, Solutions, and Open Issues, C. Turcu, Ed., ed: InTech, 2011.

[8] OASIS, "Reference model for service oriented architecture," in OASIS-Standard, ed, 2006.

[9] J. W. Walewski, Ed., D1.2 – Initial Architectural Reference Model for IoT (IoT-A Deliverable D1.2. 2011, p.^pp. Pages.

[10] "W3C Semantic Sensor Networks Incubator Group (SSN-XG)."

[11] OGC, "Open Geospatial Consortium (OGC) Sensor Web Enablement: Overview and High Level Architecture," OGC white paper2007.

[12] OGC, "OpenGIS® Sensor Model Language (SensorML) Implementation Specification," Open Geospatial Consortium, Inc2007.

[13] 52North. 52North Sensor Web Community. Available: http://52north.org

[14] J. H. Kim, K. Kwon, K. D.-H, and S. J. Lee, "Building a service-oriented ontology for wireless sensor networks," in Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science, 2008, pp. 649-654.

[15] C. D. Nguyen, A. Perini, and P. Tonella, "Ontology-based Test Generation for Multi-Agent Systems (Short Paper)," in Proc. 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AA-MAS 2008), Estoril, Portugal, 2008, pp. 1315-1318.

[16] X. Bai, S. Lee, W.-T. Tsai, and Y. Chen, "Ontology-based Test Modeling and Partition Testing of Web Services," in Proc. IEEE International Conference on Web Services, 2008.

[17] O. v. F. F. Filho and M. A. G. V. Ferreira, "Semantic Web Services: A RESTful Approach," in Proceedings of IADIS International Conference WWW/Internet, 2009, pp. 169-180.

[18] W3C, "OWL-S: Semantic Markup for Web Services ", W3C Member Submission 2004.

[19] A. Bar-Noy, G. Cirincione, R. Govindan, S. Krishnamurthy, T. F. LaPorta, P. Mohapatra, M. Neely, and A. Yener, "Quality-of-information aware networking for tactical military networks," in IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011, pp. 2-7.

[20] J. Cardoso, A. Sheth, J. Miller, J. Arnold, and K. Kochut, "Quality of Service for Workflows and Web Service Processes," Journal of Web Semantics, vol. 1, 2004.

[21] Y. Z. Tao Yu and K.-J. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," ACM Trans. Web, vol. 1, May 2007.

[22] I. Schieferdecker, Z. Dai, J. Grabowski, and A. Rennoch, "The UML 2.0 testing profile and its relation to ttcn-3," Testing of Communicating Systems 2003.

[23] "The testing and test control notation version 3 (TTCN-3)," vol. European Standard 201874, ed, 2002/2003.

[24] M. Fischer, R. Tönjes, and R. Lasch, "A New Approach for Automatic Generation of Tests for Next Generation Network Communication Services," in 6th IEEE International Workshop on Service Oriented Architectures in Converging Networked Environments (SOCNE 2011), Toulouse, France, 2011.