An Ontology-Based IoT Resource Model for Resources Evolution and Reverse Evolution*

Shuai Zhao, Yang Zhang, and Junliang Chen

State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
{zhaoshuaiby,yangzhang,chjl}@bupt.edu.cn

Abstract. In view of the characteristics of Internet of Things (IoT), the current architectures could not effectively use and manage IoT resources and information. Numerous projects in the area of IoT have proposed architectures which aim at integrating geographically dispersed and internet interconnected heterogeneous Wireless Sensor and Actuator Networks (WSAN) systems into a homogeneous fabric for real world information and interaction. These architectures are faced with very similar problems in how to support the evolution of resources and maintain service continuity, how to integrate the data which comes from heterogeneous resources. To address these issues, this paper proposes a resource model supporting dynamic evolution and reverse evolution. The resource model uses Linked Data and extends the existing ontologies, such as W3C SSN, etc. This resource model can express domain knowledge, event rules, and support event-based reverse evolution. Based on the resource model, our SOA-based framework can automatically access resources, generate and interpret semantic context information, dynamically create resources, and interpret historical data and events. The validation of the resource model and framework is shown through the CCMWS (Coal mine comprehensive monitoring and early warning system).

1 Introduction

The Internet of Things (IoT) proposed by MIT Auto-ID Center is honored as the third IT revolution following Computer and Internet [1]. The relevant theories and technologies become the research hotspot of academia. Huge amount of resources and information are provided by the IoT, but they cannot be effectively utilized in the current application mode. Some issues exist in the level of resource and information:

- Resource Access: Dynamic resource access, i.e. realizing the PnP (Plug-and-Play) of heterogeneous resources under the premise of limited resource ability and real-time service demands. Resource access includes three aspects: recognizing new

* This study is supported by "973" program of National Basic Research Program of China (Grant No. 2011CB302704, 2012CB315802). National Natural Science Foundation of China (Grant No. 61001118, 61171102, 61003067, 61132001); Program for New Century Excellent Talents in University (Grant No. NECT-11-0592); Project of New Generation Broadband Wireless Network under Grant (Grant No.2010ZX03004-001, 2011ZX03002-002-01, 2012ZX03005008-001).

C. Liu et al. (Eds.): ICSOC 2012, LNCS 7636, pp. 779–789, 2012.

resource seamlessly; interpreting and processing the information generated by the resources automatically; real-time maintaining the resource information according to the changes of resource state.

- Resource Discovery: Billions of resources provide more capabilities also bring issues in resource discovery. How to find the most appropriate resource across different platforms and networks within the acceptable time and space range?
- Resource Management: Resource management and maintenance are highly complicated due to the heterogeneity, instability and evolution of the IoT. Management framework should adapt to these characteristics to manage and track the evolution of resources; the framework should promptly perceives the changes of resource state and provides context-aware dynamic switching and scheduling mechanism of heterogeneous resource to ensure the continuity of real-time service.
- Resource Utilization: The real world can be reflected more and more realistically by the digital world with the expansion of the range and the further capability of sensing. Meanwhile, the operation and task generated in the digital world get more complicated, using a single resource is unable to be competent. How to coordinate and schedule distributed resources in a unified way according to business needs, user need context and ubiquitous resource information context?

A normative resource model and framework are the basis of addressing the issues above. Therefore, research program such as EU FP7 have been devoted to researching normative architecture of the IoT, such as PECES [2, 3], SENSEI [4], etc. This paper proposes a semantic model and resource framework which focus on the issues of resource access, resource evolution, real-world service [5] continuity, and generating context information. The model and framework have the following characteristics:

- The SOA-based framework integrates SCA (Service Component Architecture) and ESB (Enterprise Service Bus) together as the execution platform. SCA serves as the assembly standard of service components. ESB provides protocol translation, message routing, security, management functions for the interaction between components. Hot deploy and cross-platform capabilities are ensured by OSGi.
- The ontology-based semantic model can express domain knowledge and business information which are the basis of implementation the linkage (mutual influence) between model and system business logic. The model extends the current ontologies such as W3C SSN [6], OWL-S, etc. to ensure the versatility and standardization, it uses linked-data[10] to associate with functional ontologies.
- Events are generated automatically in two modes: event tree; Abductive and Deductive Reasoning (ADR). Event tree is a fast matching method and appropriate for the high real time applications. ADR is proposed by combining the improved Parsimonious Covering Theory (PCT) reasoning [7-9] with the deductive reasoning. ADR can reason out the inducement events from basic events, then forecasts what other events these inducement events can cause by deductive reasoning.
- The semantic model can be not only adapted to the evolution but also able to reverse evolution. If the model is restored to a historical state, it can restore and reproduce the real world context at that time and interpret the historical data and events. To be emphasized, the existing work only focuses on the evolution.

However, with the enhancement of real-world sensing and control ability, due to the evolution and instability of resources, tracking the evolution of the model and the capability of reverse evolution are also very important. We use graph database to store context information, use the model snapshots and archived events to track the evolution and reverse evolution.

The rest of this paper is organized as follows. Section 2 describes the semantic model and framework. Section 3 discusses the mechanism of model evolution and reverse evolution. Section 4 presents the CCMWS to validate the model and framework. Section 5 concludes the paper and discusses the further work.

2 Semantic Model and Framework

2.1 Framework

The SOA-based framework use SCA as the assembly standard and components interact with each other by service interface. The ESB provides protocol translation, message routing, security and management functions for component interaction. What's more, the underlying OSGi offers hot deploy and cross-platform capabilities.

As shown in figure 1, the framework mainly contains these components:

– Resource Adaption (RA) is used to adapt heterogeneous resources. It publishes resource description when resource accesses and wraps the Resource Interface.

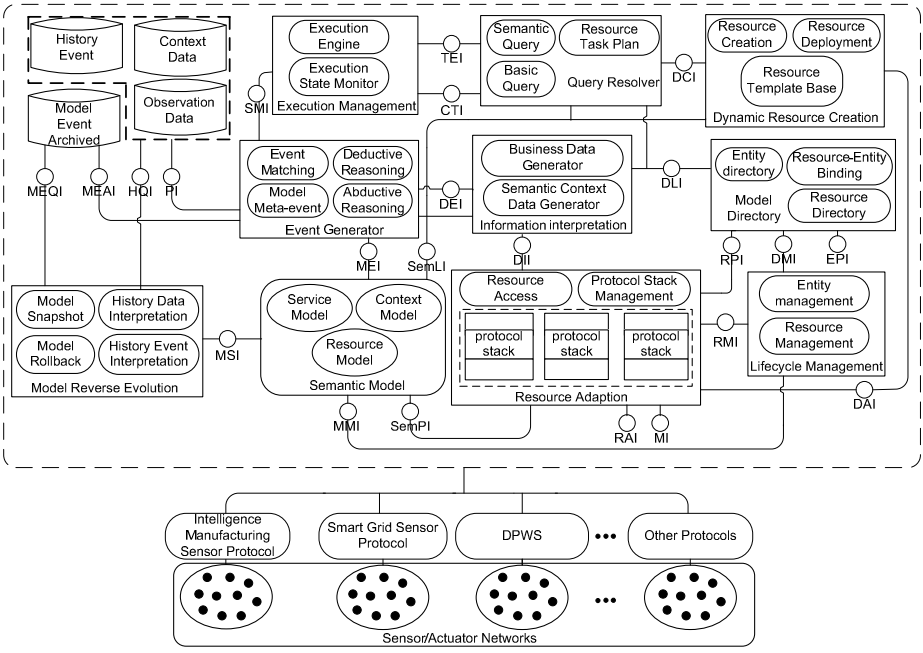

Fig. 1. Resource framework

- Model Directory, a directory of Semantic Model, is used to retrieve resources and entities efficiently.
- Lifecycle Management (LM) monitors and manages the resource lifecycle.
- Information Interpretation converts raw data to the domain data according to the definition of domain processing, promotes it to meaningful context information.
- Event Generator (EG) automatically generates events according to the event rules defined in the Context Model.
- The Persistent Component is used for persisting historical data and events.
- Execution Management (EM) executes the task plan and monitors the resource states.
- Dynamic Resource creation can create new resources based on the domain knowledge.
- Model Event Archived archives the model events using for reverse evolution.
- Reverse Evolution (RE) restores the model to a historical state based on model snapshots and events without interfering with the current model.

2.2 Semantic Model

Upper layer uses W3C SSN to describe the sensor resources and observation data, context model and domain knowledge extensions are needed. OWL-Time is used to express the time information. SemSOS O&M [11] and SENSEI O&M are extended as the observation data model. What's more, we used Linked Data to link the model to ontologies such as FOAF, GeoNames, Linked-GeoData and DBpedia[1]. OWL-S is the upper ontology for describing service. FOAF makes the Model more general and provides additional information. GeoName and Linked-GeoData are worldwide geographic location ontologies which express the global location of entities and resources. The domain ontologies in the bottom extends the ontologies mentioned above and can express domain knowledge. Figure 2 shows the Semantic Model. In order to make the figure more clearly, we removed the relationships with external ontologies.

Resource Model

Resource Model is compatible with the SSN ontology. The model is open end and can be re-classified or further classified. We use three forms to express the location of resources. They are latitude and longitude, local location and global location. Local location represents the position in the specific application scenarios. Global location links to a global position such as GeoName and Linked-GeoData. Observation Area is the resource observation or control scope. Engineering Conversion describes a conversion process which translates the raw data into available domain data. Protocol Stack describes the sensor protocol used by interacting with resources. Resource Type is a domain expert classification which is standardized and representative. It is useful in resource searching and resource-entity binding. Tag provides some information of time dimension, space dimension, capacities of the resources. Measurement Capability describes the Quality of Service (QoS) of the resources under certain conditions such as accuracy, precision, response time, delay and so on. The operation range of

[1] http://en.wikipedia.org/wiki/File:DBpediaLogo.svg

resources includes equipment life, battery life, transmission range and so on. Observation and Provide Service are another two attributes of the Resources. They link to the Service Model and Observation respectively. Observation mainly contains results, location, time, QoI and unit, etc.

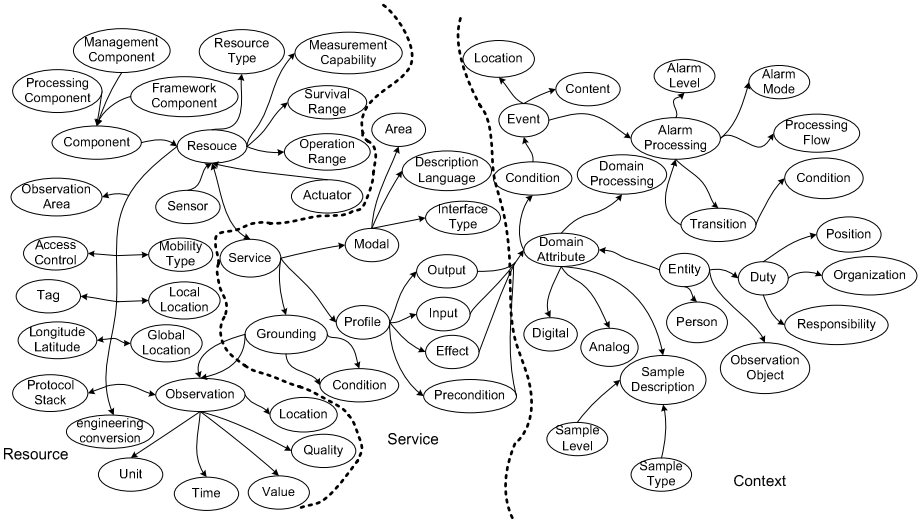

Fig. 2. Overview of the Semantic Model

Service Model

Resources are accessed by services which provide functionality to gather information about entities they are associated with or manipulate physical properties of their associated entities. Service Model describes the Real-World Service of resources which is based on the Model, Profile, Grounding concepts of the OWL-S and the service model of literature [5]. Profile has four attributes which is IOPE (input, output, precondition, effect). Input and output link to the entity attributes. Meanwhile, precondition and effect link to the entity attribute state. Model describes the service scope, service description language and service interface type. Grounding has properties such as service provider, service IOPE and so on. The input and output of the IOPE link to the Observation of Resources Model, while precondition and effect link to a SSN:Condition instance.

Context Model

Context Model includes Entity Model, Event Model and Sampling Description. Entity model consists of Person, Observation Object and Duty. Person can describe state, relations, environment information of persons. Observation Object describes the state, relationship and their relative position of the entities observed. Duty contains information such as position, responsibility and organization, etc. Event Model contains Event Tree and Alarm Disposal Process. The Event Tree makes classification on events according to location, emergency degree and theme. It also defines the relationship between events. The Alarm Disposal Process defines alarm level, alarm form, disposal process, etc. Event node associates with alarm disposal process, so the disposal process can make

appropriate decisions after the event occurs. The Sampling Description includes information of sampling level, sampling type, etc. Domain attribute is linked to sample description to define the sampling rules of attributes. What's more, it is also linked to event condition to define the trigger conditions of attribute event. In our Semantic Model, the Observation of Resource Model is associated with domain attribute of Entity Model. We call this association as Resource-Entity Binding. The binding is the basis for promoting the observation data to context information. The binding can be static or dynamic and be generated automatically or manually.

2.3 Model Evolution and Reverse Evolution

In our framework, evolution has the following aspects:

- Mobility
- State(Lifecycle, performance state, configuration)
- New resource access
- Attributes changing
- Relationship changing

These aspects are not mutually isolated but have semantic association with each other; an evolution may lead to other evolution behaviors. For example, the mobility of resource may cause resource attribute changing, resource availability changing and the binding relationship between resource and entity changing. According to these evolution aspects, there are three main model-event types:

- Lifecycle event (resource availability changing, resource access, lifecycle state changing)
- Attribute changing event (property value, state, configuration parameter and performance parameter)
- Relationship changing event(establish, update, release)

Table 1. Example of resource evolution

Time	Model Event	Model			Snapshot	Data Event
		Entity	Resource	Relationship		
T_0		E_1, E_2	R_1, R_2	$Binding(E_1, R_1)$ $Binding(E_2, R_2)$ $Closeto(E_1, E_2)$		
T_1		E_1, E_2	R_1, R_2	$Binding(E_1, R_1)$ $Binding(E_2, R_2)$ $Closeto(E_1, E_2)$		$Event$: E_1 alarm
T_2	$Event_{l1}$: R_3 $access$	E_1, E_2	R_1, R_2, R_3	$Binding(E_1, R_1)$ $Binding(E_2, R_2)$ $Closeto(E_1, E_2)$		
T_3	$Event_{r1}$: $del\_Binding(E_1, R_1)$ $Event_{a1}$: R_1 $del\_Attributes$ $Event_{l2}$: R_1 $remove$	E_1, E_2	R_2, R_3	$Binding(E_2, R_2)$ $Closeto(E_1, E_2)$		
T_4	$Event_{r2}$: $Bind(E_1, R_3)$	E_1, E_2	R_2, R_3	$Binding(E_1, R_3)$ $Binding(E_2, R_2)$ $Closeto(E_1, E_2)$		
· · · T_{Now}	…	…	…	…	…	…

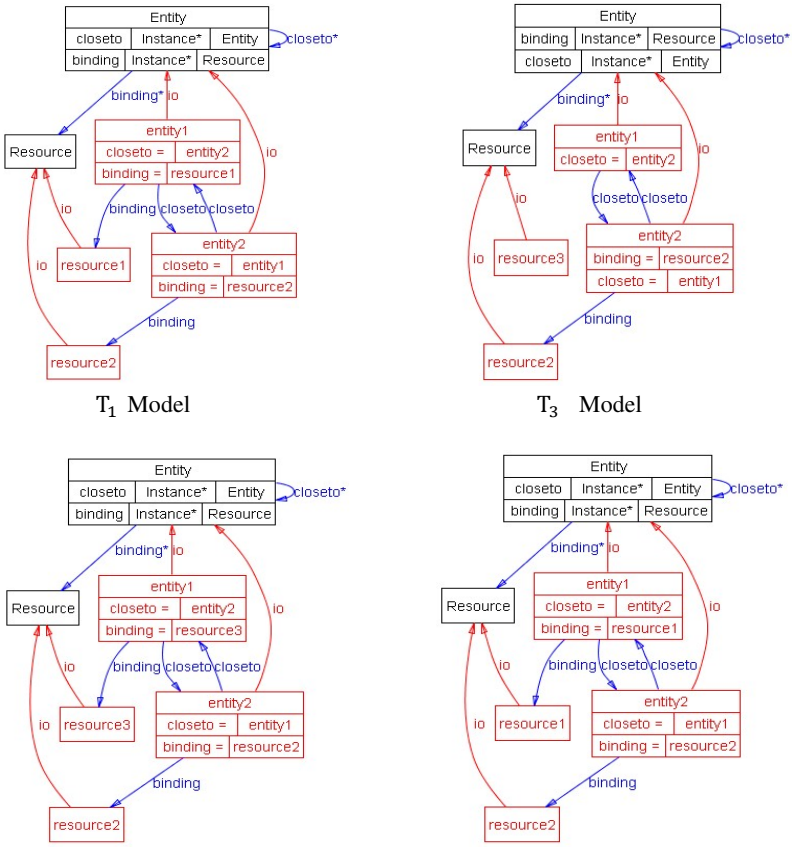

Fig. 3. Simple example of ontology evolution and reverse evolution

Model evolutions may be triggered by the underlying events from WSAN and the monitoring of resource state, or manual configurations. An evolution action often causes other evolution behaviors. For example, the sampling frequency drops lead to the performance attribute change, and the releasing of resource-entity binding if it no longer meets the requirements of entity. In order to track the evolution, EG archives the model events and Reverse Evolution (RE) snapshots the model periodically. RE rolls the model back to a historical state based on snapshots (closest to that historical moment) and events without interfering with the current model. Then, the historical model can interpret historical data and events.

Table 1 shows a simplified evolution process. Figure 3 shows the ontology which expresses the model in different time along with the evolution.

At time T_0, there are two adjacent entities E_1 and E_2; resource R_1 observes the attribute value of E_1 and resource R_2 observes the attribute value of E_2.

At time T_1, alarm event happens on E_1 observed by R_1.

At time T_2, new resource R_3 is accessed.

At time T_3, resource R_1 becomes unavailable, then "release relationship", "delete attribute" and "remove resource" events occur. All relationships and attributes related to the resource must be released before removing the resource instance.

At time T_4, assume that resource R_3 can provide observation service for entity E_1, then binding between E_1 and R_3 is established. Meanwhile, T_4 is the snapshot time.

T_{Now} is the current time.

It is unable to make a comprehensive and thorough analysis of E_1 alarm event only based on the event contents, historical model is needed. Reverse evolution mechanism can be applied to restore the model to the historical state.

For reverse evolution, system finds the snapshot which is closest to the historical time desired and then reversely does the actions in the model events which are happened in the time intervals between snapshot time and historical time. In detail, system finds the T_4 snapshot which is closest to T_1, then reversely does the actions between T_4 and T_1: releases the relationship between E_1 and R_3; adds resource instance R_1; restores the attributes of R_1; restores the binding relationship between E_1 and R_1; removes the resource R_3. So, the model can be restored to the T_1 model which expresses the context information at that time. The ontology restored to time T_1 is shown in figure 3. The domain knowledge along with the state and relationships of all resources and entities can be used to analyze historical data and events.

3 Implementation and Case Study

CCMWS (Coal Mine Comprehensive Monitoring and Early Warning System) is a coal mine safety application which is based on the resource model and framework. We integrated Apache Tuscany and Fuse ESB together as the service execution platform. With system implemented in Java, we store resource directory in H2 database, semantic model in AllegroGraph graph database and observation data in MySQL database. This section will show the function of system by two cases.

3.1 Dynamic Resource Creation

In the CCMWS, resource can be represented by graphic element vividly by relating resource instances to graphic element instances. Resource user can manage and use resources by dragging and dropping graphic element and these actions will be reflected in model automatically.

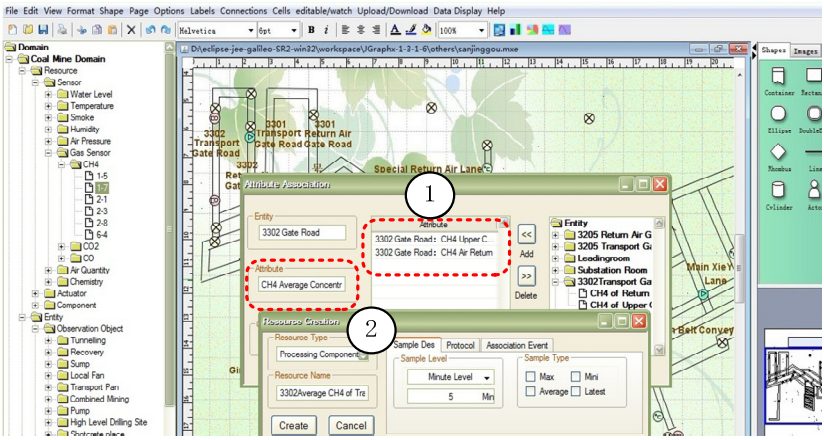

Fig. 4. Dynamic creation resource

Figure 5 shows the process of creating resource dynamically according to domain knowledge. Entity "3302TransportGateRoad" has three attributes: "CH4UpperCorner", "CH4ReturnAir" and "CH4Average". Number 1 of figure 6 shows that user defines the domain knowledge that the attribute "CH4Averagecon" can be calculated from other two attributes. Other more complicated domain processing such as "relative abundance of methane" can also be defined. "CH4UpperCorner" and "CH4ReturnAir" have been bound with resources respectively. On the basis of the conversion relationship of attributes, system can create the resource"3302TransportCH4Average" dynamically. The observation values of "CH4UpperCorner" and "CH4ReturnAir" are the inputs of the resource "3302TransportCH4Average". The resource will process the input data according to domain processing definition, the result of processing will be interpretation as the value of "CH4Average" attribute. Number 2 shows how to configure Sampling Description, Protocol Stack and Event related to the resource instance.

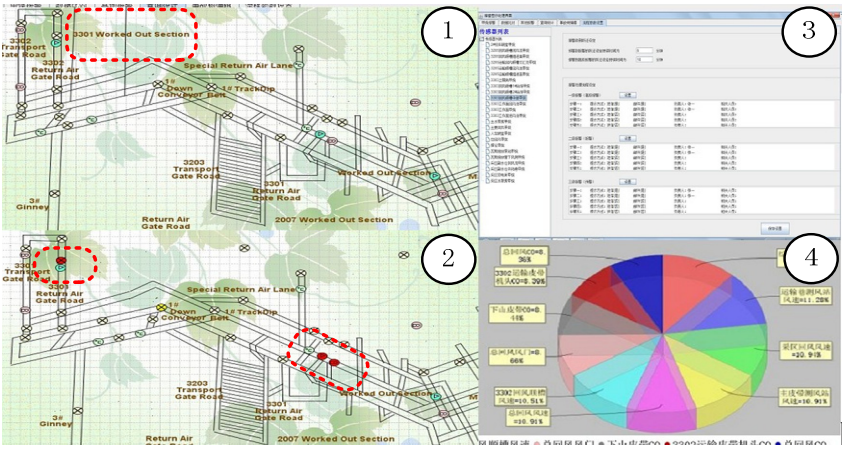

Fig. 5. Historical event analysis

3.2 Historical Event Analysis

Figure 6 shows the functions of alarm event statistics and context panorama reproduction. Number 1 displays the current panorama of coal mine, "3302" is the current working face while "3301" is worked out section. Number 4 shows the statistical result of CH4 alarm events in a period of time. According to the statistics, the highest alarm happened on "3301Transportworkingface", which reached up to 2.65%. But as the evolution of model, "3301" in current panorama has become a worked out section, the related historical state and context are unavailable in the current model. In order to analyze the event above, model reverse evolution mechanism will be applied to present the context panorama when the event happened, which can show the state of all resources and entities intuitively at that time. Then the system interprets the event based on the historical model. Number 2 shows the context panorama restored when the event happened, in which 3301 is the working face. Analysis of the historical model and panorama found that there was nothing wrong with the sensor performance and entity attribute threshold. But the "3301TransportFan" resource generated "stop working" alarm and "AirReturnLaneFeed" generated "equipment malfunction" events. So, we can determine the cause of the CH4 alarm is "power cut lead to 3301 transport fan stop working". We can also analyze the causes of "equipment malfunction" events and other hidden dangers of coal mine. Number 3 shows the analysis of the alarm disposal process executed when the alarm happened according to the historical model. The reverse evolution is very useful for failure, accident and disaster analysis.

4 Conclusion

The resource model and resource framework are the foundation of the IoT architecture, as well as the bridge between the upper layer application and the underlying WSAN. This paper proposes a SOA-based resource framework, an ontology-based resource model and proposes solutions for resource access, resource evolution and reverse evolution. There are still some aspects need to be improved. The resource-entity binding strategies need to be further studied and the binding maintainer and binding opportunity need to be further clearly defined. The current model reverse evolution method is an overall rollback which is not very effective. We plan to propose a partial rollback method based on time, space, topic and dependences, which only roll the relevant part of the model back.

References

1. Xing, L., Jin, Z., Li, G.: Modeling and verifying services of Internet of Things based on timed automata. Chinese Journal of Computers 34(8), 1365–1377 (2011) (in Chinese)
2. Villalonga, C., Bauer, M., López Aguilar, F., Huang, V., Strohbach, M.: A Resource Model for the Real World Internet. In: Lukowicz, P., Kunze, K., Kortuem, G. (eds.) EuroSSC 2010. LNCS, vol. 6446, pp. 163–176. Springer, Heidelberg (2010)

3. Haroon, M., Handte, M., Marrón, P.: Generic role assignment: a uniform middleware abstraction for configuration of pervasive systems. In: Proc. of the Pervasive Computing and Communications (PerCom's 2009), Galveston, United States, pp. 1–6 (2009)
4. Payam, B.: D3.6 Final SENSEI Architecture Framework, Public SENSEI Deliverable. CEA-LETI (2011)
5. De, S., Barnaghi, P., Bauer, M., Meissner, S.: Service modeling for the Internet of Things, pp. 949–955 (2011)
6. Semantic Sensor Network XG Final Report, http://www.w3.org/2005/Incubator/ssn/XGR-ssn-20110628/
7. Henson, C., Thirunarayan, K., Sheth, A., Hitzler, P.: Representation of Parsimonious Covering Theory in OWL-DL. In: Proc. of the 8th International Workshop on OWL: Experiences and Directions (OWLED 2011), San Francisco, CA, United States (2011)
8. Thirunarayan, K., Henson, C.A., Sheth, A.P.: Situation awareness via abductive reasoning from semantic sensor data: a preliminary report. In: Proc. of the International Symposium on Collaborative Technologies and Systems (CTS 2009), Balitimore, Maryland, USA, pp. 111–118 (2009)
9. Henson, C., Sheth, A., Thirunararyan, K.: Semantic Perception: A Semantic Approach to Convert Sensory Observations to Abstractions. In: Proc. of the IEEE Internet Computing (2012)
10. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. International Journal on Semantic Web and Information Systems (IJSWIS) 5(3), 1–22 (2009)
11. Henson, C.A., Pschorr, J.K., Sheth, A.P., Thirunarayan, K.: SemSOS: Semantic sensor observation service. In: Proc. of the International Symposium on Collaborative Technologies and Systems (CTS 2009), Balitimore, Maryland, USA, pp. 44–53 (2009)