Architecting Sensor Networks Publish/Subscribe Applications: A Semantic Technology-Oriented Approach

Biao Dong[1, a], Jinhui Chen[2, b]

[1]School of Computer & Software, Nanjing Institute of Industry Technology, Nanjing, 210023, China

[2]School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing, 210044, China

[a]email:dongb@niit.edu.cn, [b]email:cjh@nuist.edu.cn

Keywords: Sensor Networks; Publish/Subscribe; Semantic; Ontology

Abstract. This paper presents a semantic technology-oriented approach(STOA) for architecting sensor networks applications using publish/subscribe(Pub/Sub) paradigm. Considering the high level abstraction which is necessary for sensor networks knowledge sharing and reusing, sensor ontologies are defined to describe the related concepts and their relationships, both graph-based event model and graph pattern-based subscription model for sensor networks are proposed to support events and subscriptions with complex structure, and a matching algorithm is designed to detect structure matching, semantic matching and value matching. The analysis indicates that STOA uses ontology and Pub/Sub paradigm to effectively realize knowledge classification, sharing and reuse for sensor networks applications.

Introduction

Through various integrated micro-sensors, sensor networks can collaborate in real-time monitoring, sensing and gathering information on various environmental objects. In recent years, the rise of Internet of Things(IoT) has greatly promoted the development of sensor networks. However, the current sensor networks are applied to their specific fields. Because of the heterogeneity of sensor devices, data processing, communication protocols, etc., these make it difficult interconnection between sensor networks in order to effectively allocate and share resources. So, it's difficult for users to find useful information in a large number of sensor data[1]. Ontology is originally a philosophical concept. It's used by philosophers to describe the nature of things. Later, ontology is introduced into the fields of artificial intelligence, knowledge engineering, computer, and so on. The purpose of constructing ontologies is to achieve knowledge sharing, reusing, communication and interoperation between systems to a certain extent. Applications of ontology on Internet led to the birth of semantic Web. This is expected to solve the problem of semantic sharing of network information, and to realize integration of knowledge and information[2]. Therefore, the ontology-based applications play more important roles in sensor networks.

Pub/Sub is an asynchronous communication paradigm that supports many-to-many interactions between a set of clients. The loose coupling of clients eliminates the burden of context information gathering and processing by resource constrained devices and also supports reuse of context information. In addition, it hides context information access and low-level sensor operations from applications, and therefore provides context information to the applications[3]. The Pub/Sub protocol is considered as a way to achieve distributed event-driven mechanism, so this paradigm is a suitable paradigm for architecting semantic technologies-oriented approaches for sensor networks.

Esswein presented an ontology-based approach for data quality inference on streaming observation data originating from large-scale sensor networks. The approach incorporates semantic inference into a Pub/Sub messaging middleware[4]. Bröring illustrated and analyzed the recent developments of the new generation of the sensor web enablement specification framework, and pointed out challenges and resulting future work topics for research on sensor web enablement[5]. Calbimonte presented an open-source system that relies on semantic representations of sensor metadata and observations, to guide the process of annotating and publishing sensor data on the Web[6]. Jara proposed the semantic Web of Things for the integration of the semantic web on the Web of Things, and analyzed the impact of the semantic-annotations/metadata in the performance of the resources[7]. Llaves used event processing to detect event patterns in time series of observations, and represented event-related information extracted from multiples sources under a common event model. Additionally, domain knowledge is modelled in a multilevel ontology structure[8]. Su examined enabling technologies for adding semantics to the IoT, and analyzed data formats which enable IoT applications consume semantic IoT data in a straightforward and general fashion[9].

The above researches effectively improve the efficiency of system development for sensor networks, but a key problem of sensor networks semantic technology-oriented applications for further study is how to provide a unified ontology framework in order to solve problem of semantic sharing and reusing. This paper proposes a semantic technologies-oriented approach for designing and developing sensor networks information systems. Our approach supports the following features:

(1) We describe sensor ontology and inheritance relationships of sensor ontology model.

(2) We define a graph-based event model and a graph pattern-based subscription model for sensor networks.

(3) In terms of matching algorithm, we use structure matching, semantic matching and value matching.

Sensor Ontology

The meaning of sensor ontology is as follows. According to application requirements, sensor ontology is abstract description of the related concepts and their relationships. The concepts include observer, observation object, observation result, observation time, observation site, observation theme, and so on. Sensor ontology provides a clear sharing concept model for sensor data in semantic sensor Web. The goal is to explicitly express the following information, such as sensors and their characteristics, observation object, observation result, observation time, observation theme, to enhance the semantics of sensor data, and to maximize the sharing and reusing of sensor information resources. Sensor ontology has the following characteristics: sensor observation values are updated in real time, and include three aspects of contextual information: such as time, space and themes. Sensor ontology can learn from different observation areas. The concept of sensor ontology includes not only the description of its own characteristics, but also the description of its application domain and service object.

We use ontologies to represent conceptual model of sensor events. Ontologies are standard specifications for the results of the conceptualization, they describes various concepts in a certain field, the relationships between the concepts, and the constraint conditions that the concepts should satisfy. In STOA, a conceptual model of event is composed of the following parts. The first part is to describe classes and their hierarchical relationship. An entity may belong to more than one class. There may be multiple inheritance relationships between classes, but STOA does not allow the emergence of looping inheritance between classes. The second part is to describe properties and their hierarchical relationship. There may be multiple inheritance relationships between properties, but STOA does not allow the emergence of looping inheritance between properties. A Class can have more than one property, and a property can also serve multiple classes. The third part is about meta-statement. Three tuple (subject-class, property, object-class) is called meta-statement. For a given subject-class, it indicates which properties are allowed, and which object-classes the property values belong to.

Fig.1 shows inheritance relationships of the sensor ontology model used in STOA. Sensor class is defined as object which shows inheritance relationships between sensors. The sub-classes of property include survivalproperty, commonproperty, measurementproperty, and measurementrange. Among of them, commonproperty is used to represent some important properties of sensors, such as location, observation, beginning and ending time of observation. Measurementproperty includes properties such as response time, measurement accuracy, and so on. MeasurementRange contains upper and lower bounds of measurement data. In the sub-classes of quality, the information sub-class is used to define the properties which include original sensor data.

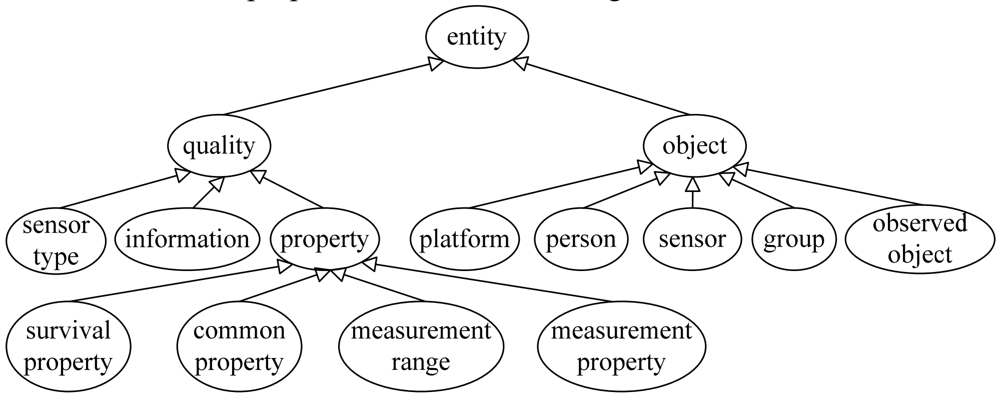

Fig.1. Inheritance relationships of sensor ontology model

Sensor Event Model and Subscription Model

Sensor event model. In sensor event model, event is represented as RDF graph which is called event graph. RDF is a way of expressing fact by three tuple (subject, property, object), where both subject and property are uniform resource identifier(URI), and object is either URI or text. The three tuple is called statement. RDF data can be represented by directed graph, where vertices and directed edges in directed graph represent subjects or objects, and properties respectively. Starting-vertex, end-vertex of a directed edge, and the directed edge itself constitute a statement, where starting-vertex is subject, and end-vertex is object of the statement. For example, we use sensor ontologies to describe a set of weather data, its event graph is shown in Fig.2.

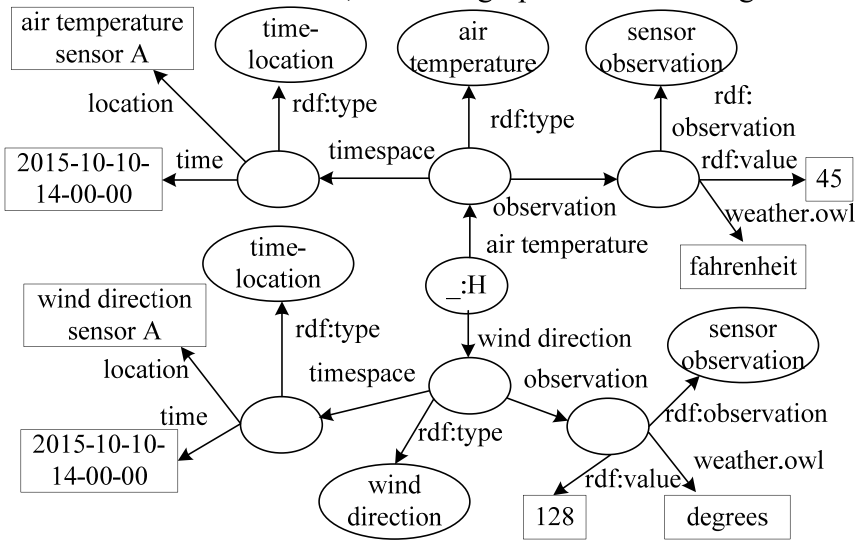

Fig.2. An event graph

In order to facilitate processing event graph, we make the following restrictions: in event graph, there is only one vertex, which is called master vertex. There is a path from the master vertex to any other vertexes.

Subscription model. In STOA, because event is represented as RDF graph, the user's subscription is actually graph pattern built on the syntax of the RDF graph. The graph pattern defines the shape of the graph as well as the constraints to vertices and edges. We designed a STOA subscription model. In the subscription model, several statement patterns use 'and' operator to form a subscription. A sentence pattern describes a statement in event graph, its form is as follows: (subject, object, meta-statement, [filter_func(object)]). Among them, the subject and object parameters are used as subject and object of sentence. They can be specific values or variables, the variables can be matched with any particular value. The meta-statement in sentence pattern specifies a type constraint that should be met. The filter function filter_func(object) is a Boolean expression that further limit the object variables.

In STOA, each subscription is represented by a graph, which is called subscription graph. In subscription graph, a vertex is marked as (id, type, [filter_func(id)]), and corresponds to a vertex in event graph, where the parameter id is a variable name or a vertex in event graph, and the parameter type is the corresponding class for the parameter id. When a vertex of the corresponding event graph is a text vertex, the sentence pattern can have an item filter_func(id), which is used to indicate which constraints the parameter id should satisfy. A tag on directed edge is a property name. A meta-statement is composed of a property name, a type of a directed edge's starting-vertex, and a type of a directed edge's end-vertex. Furthermore, a sentence pattern is composed of a meta-statement, IDs of a directed edge's starting-vertex and end-vertex, and a filter function.

Matching Algorithm

In STOA, if an event matches a subscription, the following conditions should be met. First, the property set of subscription is a subset of the property set of event. Second, for at least one event class in all event classes, there is one of the following relations, such as subclass, equivalence and equality, between the event class and subscription classes. Finally, a property value in an event satisfies the constraint defined on the corresponding subscription property. Therefore, matching of subscriptions and events in STOA actually includes three aspects: structure matching, semantic matching and value matching. Structure matching is similar to traditional vector matching, semantic matching is the matching of semantic information introduced by ontology reasoning, and value matching is the lowest level of matching.

Structure matching. Structure matching is the first step of the matching algorithm. For a given event, structure matching is to find all subscriptions that match the event in structure. The steps of structure matching are as follows.

(1) Subscriptions and events are read into a subscription index table and an event index table, respectively.

(2) An item of class vector is taken from the event index table. If the item is not null, the properties to which the item point are searched from property vector, and the positions of the properties in the subscription index table are recorded in a position set. If the position set is null, go to step 4.

(3) An item of position set is taken from the event index table. If the item is not null, from the subscription index table, the subscription vectors to which the properties corresponding to the position of the item point are searched. If the subscription property set is a subset of the position set, then the registration numbers of subscribers are taken out from subscribe vectors, and are added in structure matching subscription set. The algorithm don't judge the next subscription vector until all of the subscription vectors to which the properties point have been accessed. And then repeat the step. If the item is null, go to step 2.

(4) Return the structure matching subscription set.

Semantic matching. Semantic matching completes semantic reasoning of a given class, and filters the result of reasoning. Semantic reasoning here mainly refers to reasoning ancestor relationship of class. In STOA, semantic matching is divided into two parts: semantic matching of event type and semantic matching of property value type. The essence of these two parts is about the relation reasoning of types. These two parts have some differences only in specific implementation process. The basic matching process is as follows.

(1) Ontologies are read into specified model. We use adjacency list to store ontology graph.

(2) The algorithm obtains all ancestors of a specified class from the adjacency list, and gets a set of a specified class and its ancestors. The set is denoted as S.

(3) Determine whether subscription constraint values belong to S. If true, it indicates that the event meets the constraints. The algorithm continues to judge the next constraint. Otherwise, the algorithm quits the subscription, and continues to match the next subscription.

Value matching. Value matching is to check whether a property value of an event satisfies a given property constraint in a subscription. Value matching consists of two parts: string matching and semantic value matching. If a property in subscription belongs to data property in ontology, matching of property constraint value is string matching. If a property in subscription belongs to object property in ontology, matching of property constraint value is as follows. First, all ancestors of the property constraint are obtained according to ontology definitions. Second, string matching is performed to check whether event property values are contained in the property constraint value set of the subscription and its ancestors. In a process of value matching, if an event is able to satisfy all properties in a subscription, the subscription is a successful subscription to the event. The steps of value matching are as follows.

(1) Read a property in a subscription, and determine whether the constraint value of the property is equal to the value of the corresponding property in an event. If true, repeat this step until all property constraints of the subscription have been compared. Otherwise, go to step 2.

(2) Read ontologies into a specified adjacency list.

(3) Get a property of the subscription, and determine whether the property belongs to object property by the adjacency list. If true, go to step 4. Otherwise, quit the subscription, and continue to match the next subscription.

(4) Read the event property values, find out all of its ancestors according to the adjacency list, and determine whether the value of the subscription is equal to one of its ancestors. If true, go to step 1. Otherwise, quit the subscription, and continue to match the next subscription.

Conclusion

In this paper, we propose STOA for architecting sensor networks applications using Pub/Sub paradigm. We derive conclusions of event and subscription models from analysis, and design a matching algorithm from three aspects: structure, semantics, and value. The results show that STOA effectively realizes knowledge classification, sharing and reuse for sensor networks applications.

Acknowledgement

This work was sponsored by Qing Lan project(Jiangsu province, china).

References

[1] D.Villa, F.Moya, F.J.Villanueva, et al. Ubiquitous Virtual Private Network: A Solution for WSN Seamless Integration[J]. Sensors, 14(1), 2014, pp.779-794.

[2] Staab, Steffen, and S.Rudi, eds. Handbook on ontologies. Springer Science & Business Media, 2013.

[3] E.Souto, G.Guimarães, G.Vasconcelos, et al. Mires: a publish/subscribe middleware for sensor networks[J]. Personal and Ubiquitous Computing, 10(1), 2006, pp.37-44.

[4] S.Esswein, S.Goasguen, C.Post, et al. Towards ontology-based data quality inference in large-scale sensor networks[C]. Ccgrid 2012 IEEE Computer Society, 2012, pp.898-903.

[5] A.Bröring, J.Echterhoff, S.Jirka, et al. New generation sensor web enablement[J]. Sensors, 11(3), 2011, pp.2652-2699.

[6] J.P.Calbimonte, S.Sarni, J.Eberle, et al. XGSN: An Open-source Semantic Sensing Middleware for the Web of Things[C]. 7th International Workshop on Semantic Sensor Networks, 2014.

[7] A.J.Jara, A.C.Olivieri, Y.Bocchi, et al. Semantic web of things: an analysis of the application semantics for the IoT moving towards the IoT convergence[J]. International Journal of Web and Grid Services, 10(2-3), 2014, pp.244-272.

[8] A.Llaves, W.Kuhn. An event abstraction layer for the integration of geosensor data[J]. International Journal of Geographical Information Science, 28(5), 2014, pp.1085-1106.

[9] X.Su, J.Riekki, J.K.Nurminen, et al. Adding semantics to internet of things[J]. Concurrency and Computation: Practice and Experience, 27(8), 2015, pp.1844-1860.