Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2014, Article ID 693957, 13 pages
http://dx.doi.org/10.1155/2014/693957

Research Article

Semantic Reasoning with Contextual Ontologies on Sensor Cloud Environment

Kyungeun Park,[1] Yanggon Kim,[1] and Juno Chang[2]

[1] *Towson University, 8000 York Road, Towson, MD 21252-0001, USA*
[2] *Sangmyung University, 20 Hongjimoon-2-Gil, Seoul 110-743, Republic of Korea*

Correspondence should be addressed to Kyungeun Park; kpark@towson.edu

Received 24 December 2013; Accepted 24 March 2014; Published 28 April 2014

Academic Editor: Yujin Lim

This research first focused on processing enormous number of sensor events from a variety of city-wide sensor networks. As a solution, the well-known big data handling scheme, Hadoop cluster framework, drew, unquestionably, our attention. The acquired sensor events are to be used to immediately detect a certain abnormal situation within the framework. Accordingly, we integrated our existing context-aware collaboration framework with Hadoop cluster framework by interfacing data collection and context-aware reasoning parts of the existing framework with the Hadoop cluster framework. This approach enabled us to effectively process massive sensor events and semantically analyze the big data within the cluster environment. The proposed smart city sensor cloud framework provides ontology-enabled semantic reasoning scheme with the XOntology in combination with the Context-Aware Inference (CAI) model. By applying the ontology technology, the proposed framework enhances the availability and interoperability of the contextual information across many cooperating parties according to semantic reasoning results. Further, this framework is flexible enough to integrate any heterogeneous platforms including many existing IT solutions as well as mobile platforms. In addition, this approach presents the direction of progressive migration of many existing sensor network solutions into big data handling sensor cloud framework.

1. Introduction

Now sensors are everywhere around us. With the rapid progress in wireless telecommunication and multidevice sensor technology, wherever you go, you can expect any kinds of sensors waiting for you. They are to work to assist the traditional way of our lives by providing most of the existing IT solutions with their smart computing power and innovative services in ultimate smart city environment [1]. This results in producing explosively increased sensor data, meaning that we need to find a new data handling method to effectively process them and encourage various IT systems to collaborate with each other through increased information sharing for more accurate and pertinent services. On the other hand, it is true that we have to find an effective way to keep rapidly growing data volume to be ready for use at any place in real-time. As a solution, the well-known Hadoop [2] big data cloud framework is considered to support huge events data. The Hadoop based big data cloud framework, however, has latency limitation in processing the huge data due to its batch-oriented data processing mechanism [3]. In a sense, this makes it difficult to constantly monitor and immediately respond to a certain emergency situation or a specific attention required situation. Most importantly, the anticipated services for a smart city environment should deal with an urgent situation with agility as well as normal conveniences for citizens. This requires the framework to monitor the events and to detect abnormal situation among tons of incoming events.

As a solution, the existing ontology-enabled collaboration framework is combined with the Hadoop cloud framework to build a complementary cooperating system. The existing framework was first implemented as a scenario-based collaborative framework where numerous heterogeneous application services collaborate with each other according to the predefined service scenarios depending on specific event(s) from sensor network environments [4]. Particularly, rapid progress in wireless telecommunication and RFID/USN

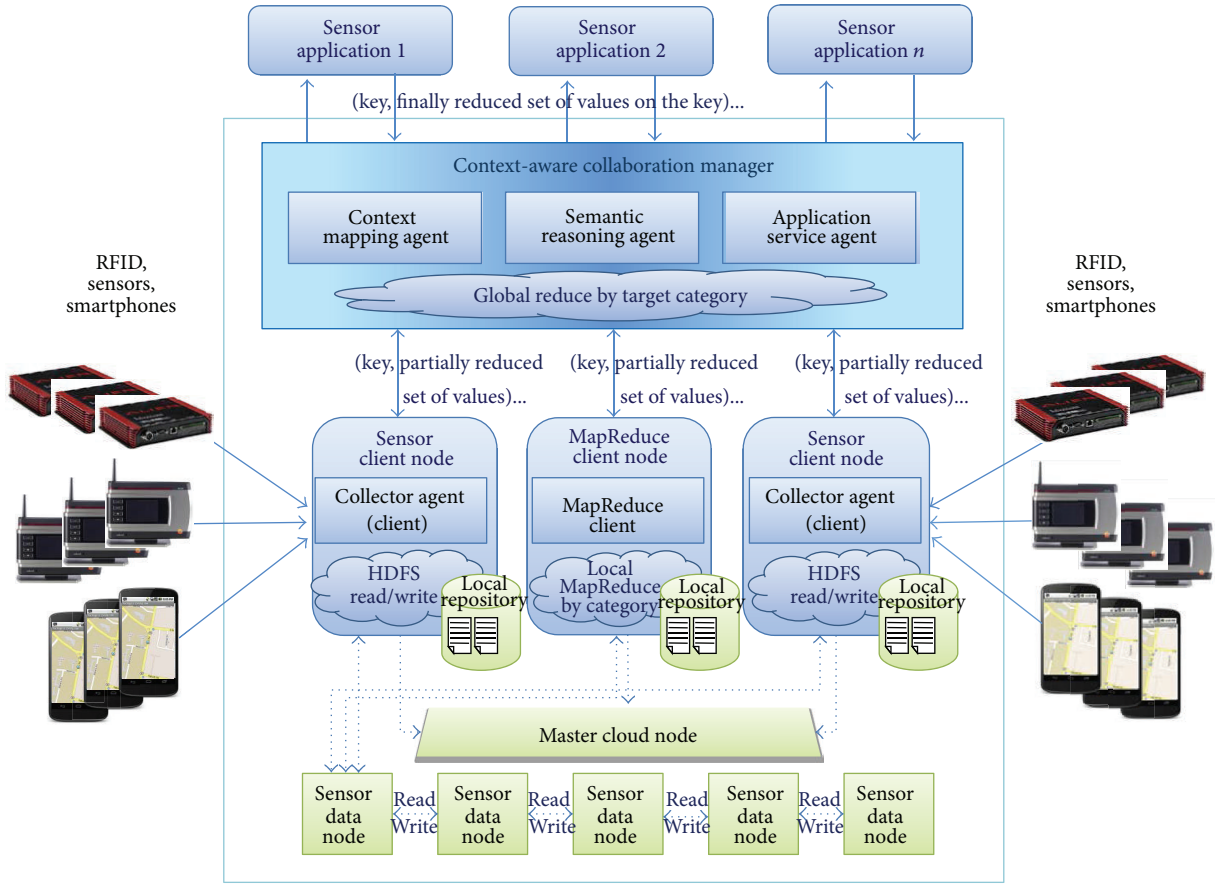

Figure 1: Smart city sensor cloud framework.

sensor technology encouraged us to apply this advanced smart computing power to the existing IT solutions. The initial collaborative RFID/USN-enabled adaptive middleware framework has been extended to perform semantic analysis based on associated contextual data from numerous sources. For this, Context-Aware Inference (CAI) model with the contextual ontologies is applied to the framework as the common semantic analysis infrastructure [5–7].

Figure 1 describes the overall architecture of the smart city sensor cloud framework where a variety of sensor-enabled services are supplied depending on contextual information resulting from the semantic reasoning within the integrated framework. The proposed framework includes Hadoop Distributed File System (HDFS) [2] as the main big data handling scheme of the sensor cloud infrastructure and the context-aware collaboration manager as the semantic analysis controller. The sensor cloud infrastructure contains a single master cloud node, sensor client nodes, and up to thousands of sensor data nodes. Especially, a collector agent is built on the sensor client node as a HDFS client and performs user-customized map and local reduce functions as a MapReduce [8, 9] client. Incoming events streams are first saved on the real-time stream repository on the sensor client node and replicated on the pipelined sensor data nodes.

The collector agent in the front-end of each sensor client node not only writes the events into the sensor cloud as a HDFS client [2] but also frequently asks MapReduce functions to be executed on the newly accumulated events on the associated data nodes. In addition, the proposed framework takes full advantage of spatiotemporal locality with the customized MapReduce functions on the contextual events. Generally, semantic reasoning of the sensor events is requested by the context mapping agent to the semantic reasoning agent with the results of MapReduce functions which are applied to the events stored within the sensor cloud framework.

The context-aware collaboration manager mediates the flow of events from their acquisition at sensor client node to their consumption by application services through its semantic reasoning procedures. The context mapping agent first sorts out valid contextual information among numerous incoming events with global reduce functions and determines whether the reduced results need to be semantically observed by the semantic reasoning agent.

The previous ontology-enabled context-aware collaboration framework includes semantic reasoning scheme over continuously incoming contextual events streams that can trigger application services with proper information. Basically, the application services collaborate with each other by sharing contextual information across many involved parties. In order to provide end-users in a smart city environment with the highly sophisticated services of interconnected

applications, we need to consider an orchestrating service framework performing comprehensive analysis about the acquired contextual information. This requirement results in integrating ontology-enabled semantic reasoning scheme within the context-aware collaboration manager which adopts many features from the initial scenario-based collaborative framework [4–7].

Furthermore, smartphones allow the users to stay connected to the Internet at all times, and the pervasiveness of the sensor devices gives them the opportunity to use the information from the devices. These factors increase the need to process a huge number of data transactions and infer the present situation from the current context. The factors that are included in the current context contain the owner's current location and time, status values of various sensors, media preferences, community accessibility, and so on. This new environment leads to the development of personalized context-aware services, too.

With the introduction of the semantic reasoning agent, the applications work more efficiently and comprehensively than before because each service is defined by much finer context-driven scenarios. For example, when a handicapped passenger with a RFID-embedded boarding pass is approaching a boarding gate, the airline employee is already aware of his/her arrival and is going to be ready to service required assistance for the passenger. The semantic reasoning agent has been evolved by strengthening ontological reasoning features in the context-aware collaboration framework. In addition, the semantic reasoning agent includes the ontology definitions which define the entities of real-world objects and their properties in order for the application services to share the concepts and relationship information. The adoption of ontology into the framework results in enhancing the interoperability and system reusability. The semantic reasoning agent includes the XOntology as a context representation scheme which is shared by related application services. The integration of an ontology scheme not only allows applications to share contextual information based on the common ontology specifications but also increases the collaboration rate between application services. The interoperable XOntology is composed of core and extended ontology sets. The core ontology includes person, place, and time information. The extended ontology covers high-level applied concepts such as inventory management, logistics, surveillance function, and environmental monitoring.

This research includes the CAI model [5–7] specialized to the ontology-enabled context sharing across the context-aware collaboration framework. This option is made on the basis of the general situation analysis in combination with the ontology specification manipulating the real-world entities and their relationships.

This paper is composed of 5 Sections in total. Sections 1 and 2 present the introduction and the related research. Section 3 describes the sensor events handling and semantic reasoning schemes on the sensor cloud framework. In addition, the XOntology in terms of smart airport environment is discussed. In Section 4, the CAI model is described based on contextual situation analysis. Section 5 concludes with future research direction.

2. Related Research

The demands for handling rapidly increasing contextual events stream and their interoperability through context sharing leads us to study adopting the following areas: big data Hadoop framework; context-awareness; interapplication context sharing; and context manipulation with ontologies.

2.1. Hadoop Big Data Framework. As the well-known big data framework, Apache Hadoop software library allows for the distributed processing of large data sets across clusters of computers [3]. The framework is mainly composed of Hadoop Distributed File System (HDFS) for providing high-throughput access to application data and MapReduce scheme for parallel processing of large data sets. On the other hand, Hadoop YARN or MapReduce 2.0 works as a framework for job scheduling and cluster resource management. The two major functionalities of the JobTracker, resource management and job scheduling/monitoring, are handled by separate daemons, a global ResouceManager (RM) and per-application ApplicationMaster (AM). In this architecture, the ResourceManager controls resources for all the applications in the system by allowing per-node slave, the NodeManager (NM), to form the data-computation framework, whereas the per-application ApplicationMaster works with the NodeManager(s) to execute and monitor tasks. The ResourceManager has two main components: Scheduler and ApplicationsManager.

2.1.1. HDFS. HDFS is a distributed file system and a framework for the analysis and transformation of very large data sets using the MapReduce paradigm [8]. Typical HDFS is composed of a single NameNode for each cluster and hundreds (and even thousands) of DataNodes. The NameNode holds file system metadata in memory and keeps track of the namespace tree and the mapping of file blocks to DataNodes. Each DataNode performs a number of Hadoop application tasks on its data. User applications access the file system using the HDFS client library such as read, write, and delete functions.

2.1.2. MapReduce. MapReduce is a programming model and an associated implementation which are automatically parallelized and executed on large clusters of commodity machines. MapReduce offers easy to use programming scheme, even for programmers without experience with parallel and distributed systems, since it hides the details of parallelization, fault tolerance, locality optimization, and load balancing that are entirely transparent to the programmers [9, 10].

Hadoop divides each MapReduce job into a set of tasks [11]. Each chunk of input is first processed by a map task and converted to the sequence of ⟨key, value⟩ combinations as the result of customized map functions. After completing individual nodes' map function, all the results are reviewed and merged upon key values with the user-provided reduced function [2].

2.2. Context-Awareness. In the earlier stage of context-aware technology, many people wanted to define their own meaning

of "context," and many types of variation of "context" were found. The term, "context-aware," was introduced in Schilit and Theimer (1994) [12] for the first time. The authors describe context as location, identities of nearby people and objects, and changes to those objects. Three years after, Ryan et al. (1997) [13] referred to context as the user's location, environment, identity, and time. After that, one of the most accurate definitions is given by Dey and Abowd (2006) [14], who regard context as "any information that can be used to characterize the situation of entities (i.e., whether a person, place, or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves" [15].

As various kinds of event data and services interact with each other in an environment, a single event can relate to numerous services, and several different events may have to do with the same specific context. This means that the framework can link interconnected environments between physical devices and application services. Also, we realized that original events and the identified context data which exist within the framework should be mutually understandable by related parties. As a result, an ontology scheme is additionally applied as mutual communication method in our previous context-aware collaboration framework. This method not only allows applications to share contextual information based on the common ontology specifications but also provides a powerful way to develop adaptively collaborative services.

2.3. Interapplication Context Sharing. Since an Internet connection is available at any time and any place, we have faced an increasing demand to take full advantage of this highly intelligent environment where multiple data sources are available. Particularly, the demand leads us to develop a ubiquitous infrastructure framework in which all the context data collected from various sources is managed and shared across the correlated applications of all interested parties.

First of all, the underlying framework needs to understand various kinds of context information according to a common context description. This is done in order to enhance the effectiveness in representing and sharing the information. Most importantly, the common context description scheme needs to be clear and general enough to be accepted by all related parties.

As a solution to representing contextual information, RDF (Resource Description Framework) [16–19] and OWL Web Ontology Language [16, 20–23] of the Semantic Web standards are applied to the common context description scheme of the framework. According to the World Wide Web Consortium (W3C), "the Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries." The ultimate vision of the Semantic Web is to make automated semantic agents access the Web on their own and carry out tasks only by referring to the semantics encoded into the Web page on behalf of users [24].

In the Semantic Web, we regard the things in the world as resources: a resource can be anything that someone might want to talk about and that can be easily understood as a thing or an entity [25]. In current environment, a user's current location, real-time traffic information, environmental information like temperature, bioinformation like one's blood pressure and glucose level, and a RFID tag attached to a traveler's luggage are all examples of things to be focused on.

2.4. Context Manipulation with Ontologies. The current ubiquitous computing environment contains lots of smart objects working with their corresponding application services. This phenomenon requires us to recognize the current situation out of the complicated combination of contextual information. In addition, the application services need commonly understandable scheme to share the contextual information in this environment. First of all, the sharing of current context begins with the clear definition of common vocabularies. Once we build the dictionary of the common vocabularies, we further proceed to detect a situation determined by the individual or combined contextual information. This process is treated by adopting an ontology scheme as a tool for denoting the contextual information and describing the relationships among the individual entities.

2.4.1. Ontologies. Ontology has been described as "a specification of a conceptualization," by Gruber in 1995 [21]. He mentions that "what "exists" is exactly that which can be represented." In other words, all the intelligent equipment and objects in the current computing environment should be represented in a certain ontology language and shared by various parties. In the real world, especially Semantic Web world, ontology is about "the exact description of things and their relationships" [26].

A general ontology model describes concepts of target domain, properties and attributes of these concepts, and constraints on properties and attributes. Additionally, it optionally examines individuals, defining a common vocabulary and encouraging a shared understanding [27].

On the other hand, Neches et al. emphasize the importance of ontology as a method to enhance knowledge sharing and reusability across different applications [28, 29]. Once ontologies for a specific domain have been built, it can be shared and reused for other domains by adding new concepts into the existing one.

As a domain modeling tool, ontology languages such as RDF (Resource Description Framework), RDFS (RDF Schema), and OWL Web Ontology Language of the Semantic Web standards describe the common vocabularies and their relationships within the framework.

RDF is a primary representation language for domain models or ontologies. It allows us to define things, namely, resources as well as their properties, the relationships between resources [14, 17, 30]. RDF is composed of triples (statement), subject-predicate-object, and the formation of a graph of the resources [31]. RDF serves as a general-purpose language for representing information in the Web, whereas RDFS includes the vocabulary of RDF [19]. RDFS is introduced to strengthen the semantic capability of RDF. In other words, it provides a syntactic specification mechanism to allow us to describe resources as classes and their relationships as properties.

In OWL, a class denotes a group of individuals characterized by a certain common attribute, and it contains the set of objects. While modeling a domain of interest, interclass relationships are described as a set of subsumption (super class-subclass hierarchy) relations, a collection of superclass-subclass relationships [32]. These relationships are represented as a class hierarchy. In a similar way, a hierarchy of properties, attributes, or relationships is formed to a property hierarchy in OWL scheme [20, 23].

In Section 3, the XOntology of a smart airport prototype environment is introduced with the semantic representation of the domain by using the Protégé [33] ontology development tool.

2.4.2. Reasoning Out of Formal Ontologies. Knowledge is represented in the form of ontologies since machines interpret and express concepts, relations, and specifications. The knowledge base is extended by deriving new facts from existing ontologies. Knowledge can be formalized by using knowledge interchange format (KIF) [21, 34], conceptual graphs [35, 36], or description logics (DL) [17, 37, 38]. Among them, description logic is selected to formally represent concepts because it is expressive enough to describe logics of realistic applications. Usually, a reasoner checks consistency of the T-Box of terminology which describes terminology and rules and the A-Box of assertions, which looks at assertions about individual concepts. It also checks subsumption relationship across the existing relations [37–40]. This scheme is integrated into the existing collaboration framework as an ontology gateway, the semantic reasoning agent, which can be used as the ontology-enabled context-aware knowledge option.

2.4.3. Semantic Queries Using SPARQL. In this paper, context sharing across collaboration framework is supported by defining context ontologies as XOntology and reasoning contextual information by querying the RDF triples of the XOntology through SPARQL (SPARQL Protocol and RDF Query Language) API [41, 42]. This approach is expected to increase the interoperability and availability of the contextual information which encompasses tag, location, USN sensor, behavioral, and alert information of the constituent individuals of the XOntology. Semantic queries are supposed to find basic relationship among a variety of instances of the classes and derive logical anomalies from the current context according to the semantic rules of the ontology across collaboration framework. Depending on the query result, the framework triggers related services and enables individual service providers to provide ontology-enabled context-aware services.

3. Sensor Cloud Framework

Generally, a typical sensor event contains its spatial reference that could help find the place of its occurrence with the best available positioning method and its timestamp. Even though each event is stored into the separate sensor data node based on HDFS data replication policy, a block as a unit of a data chunk naturally contains sensor events from physically close sensors within narrow time period. This leads us to focus on effective use of the sensor cloud framework by fully making use of its spatiotemporal locality and conferring the benefit to many sensor-enabled services throughout the whole framework.

3.1. Sensor Events Handling on Cloud. In the proposed sensor cloud framework, any events from sensors are first acquired by corresponding sensor client nodes (SCNs) and replicated on many sensor data nodes (SDNs) without further intervention from the master cloud node (MCN) once the distributed file system set-up has been established. The MCN exists per cluster and engages in maintaining the mapping of the sensor clients and multiple SDNs (see Figure 1).

Normally, the sensor cloud framework requests the individual SDNs to execute map functions over the events on them from diverse perspectives such as their location, sensor types, or tag identification numbers. The initially classified events by map functions are then locally reduced according to the major categorization factor such as timestamp, sensor values, or regions as the optimization method. Then the intermediate sets of combination of key and partially reduced set of values are aggregated by the global reduce function by the context-aware collaboration manager depending on sensor applications.

In this research, the existing collaboration framework is combined with the Hadoop Framework [3] to make the framework scale and fault-tolerant across the numerous nodes as well as make use of concurrently executed MapReduce functions on distributed sensor data nodes. Before we further proceed to explain main functionalities of the sensor cloud framework and the associated applications, the categories of the sensor events for smart city environment need to be examined: RFID or any identifiable tags; sensor device events; and social message events. Each event generally contains its id, the best available spatial reference information, and the correspondent timestamp. The accuracy level of the spatial reference information varies from GPS-level to Wireless base station-level or to RFID Antenna-level quality depending on the type of sensor devices and the current environment.

MapReduce functions can be customized to reduce the latency and maximize the parallelism at a massive scale large cluster environment. As a solution to this requirement, spatiotemporal data processing approaches are combined with the typical way of alpha-numeric data handling in Hadoop based distributed data processing framework. Particularly, the nature of spatiotemporal locality at the same cluster could enrich the performance of the framework. First the map functions can be classified into MapByLocation, MapBySensor, and MapByTag depending on the nature of the events and target services. The subsequent reduce functions include ReduceByTimestamp, ReduceByContext, ReduceByRegion, ReduceByValue, and ReduceByTopology. Among them, ReduceByContext adopts the ontology-enabled semantic reasoning scheme with predefined ontologies combined with the Context-Aware Inference (CAI) model of the existing collaboration framework. Based

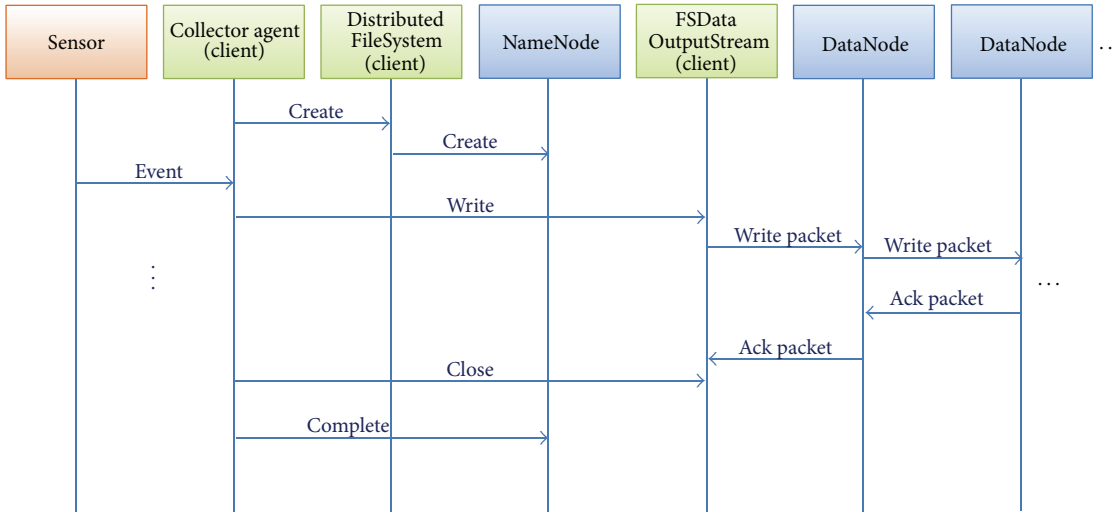

Figure 2: Sensor event to sensor cloud infrastructure.

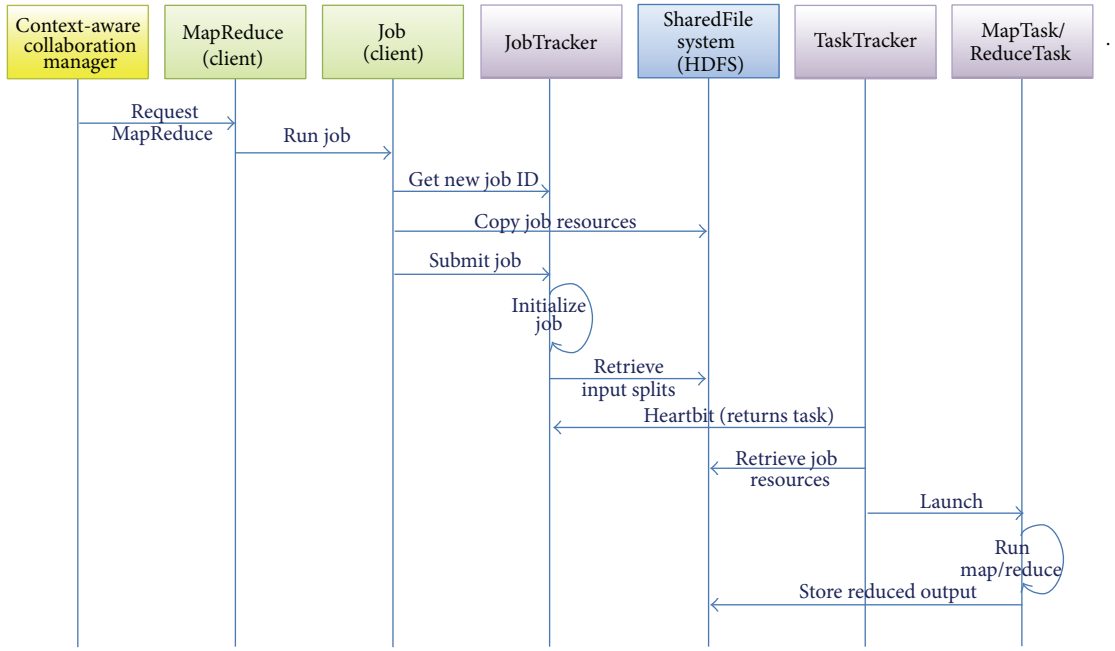

Figure 3: MapReduce client on sensor cloud framework.

on the properties of the ontologies and behavioral activities of individual event instances, the ReduceByContext function semantically evaluates the current contextual information and responds on any unexpected situations according to the semantic reasoning results.

The following two figures introduce the sequence between class entities across the whole sensor cloud framework. Figure 2 describes how the sensor clients save the acquired sensor event on DataNodes of the sensor cloud infrastructure (HDFS). Figure 3 shows how the MapReduce clients interact with the context-aware collaboration manager and the sensor cloud framework. The MapReduce function requested by the context-aware collaboration manager will be run on DataNode(s) as individual tasks and globally reduced to produce the finally aggregated result.

3.2. Semantic Reasoning. For semantic reasoning of sensor events, the context mapping agent determines whether an event (fact) is contextual information associated with the predefined properties of the XOntology. The semantic reasoning agent, then, queries RDF triples stored in the ontology repository by using Jena SPARQL API [42] and query engine. It also evaluates current context as either normal or abnormal state after reviewing if it violates the predefined situation analysis (SA) rule of the CAI model.

Figure 4 depicts the component and event flow of the semantic reasoning process of the context-aware collaboration manager. This allows the ontology-based applications to build and share the XOntology model with common vocabularies and flexible contextual reasoning within the framework.

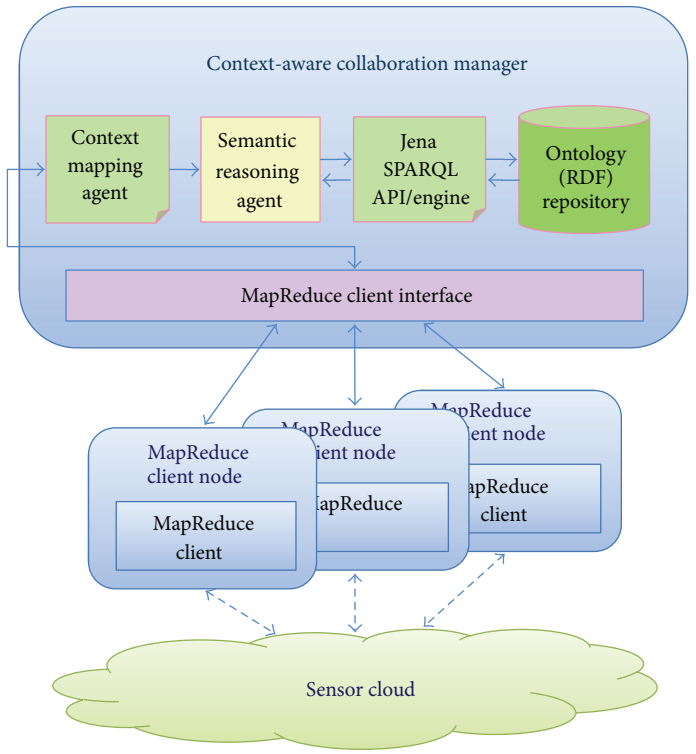

Figure 4: Context-aware collaboration manager.

3.3. XOntology. The context representation of the framework is described by the XOntology, which is composed of the core and extension ontologies. By defining the common ontologies, identified context is understood and shared between the related parties of the framework. In addition to defining the common concepts as XOntology classes, the relationships between the ontology classes are also defined as properties (see Figure 5).

The core ontology classes encompass person, place, and time concepts, and the extension ontology describes high-level concepts that depend on domain specific concepts or entities. The example of ontology describes the ontology hierarchy of an application domain of the smart airport mobile security control system, which includes agent, aircraft, cargo, event, IT systems, mobile device, public transportation, security device, service, and vehicle. The individual classes are not only drawn by analyzing real-world airport management rules, regulations [43], and the latest airport IT solutions [44], but they are also derived by combining general security issues in the computer security handbook of NIST (National Institute of Standards and Technology) [45].

The XOntology is defined by using the Protégé ontology development tool, where the ontologies written in the Protégé are exported into a variety of formats including the RDF(s), the OWL, and the XML Schema [30, 33, 46].

3.3.1. Ontology Classes. The constituent subclasses of the core and extension ontology classes in the XOntology scheme are described in Tables 1 and 2 [30, 33, 46].

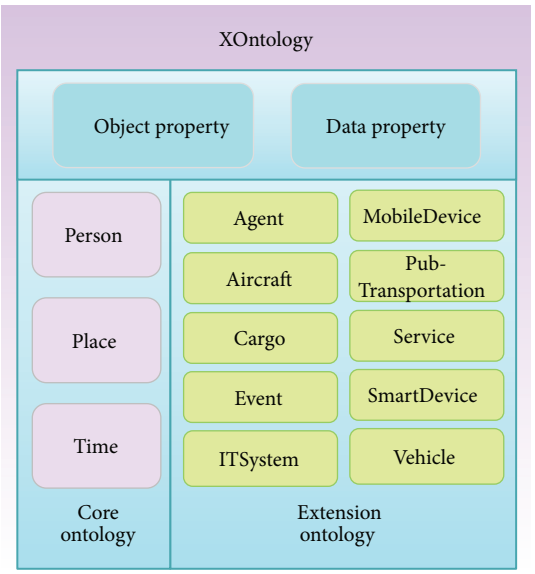

Figure 5: XOntology with core and extension ontology.

3.3.2. Ontology Properties. Relationships between classes and attributes of a member of a class form property hierarchy in the XOntology scheme. The properties are classified into the object properties and the data properties (see Table 3).

3.3.3. XOntology Modeling with Protégé. This section explains the XOntology modeling specifications designed on Protégé.

Table 1: Core ontology classes.

Classes	Subclasses
Person	Passenger, Employee, FlightCrew, Contractor, and so forth.
Place	Terminal, CheckIn, SecurityCheck, PassportControl, Gate, ImmigrationCheck, BaggageClaim, Custom, SecurityIDArea, Lounge, ParkingLot, AirCargoHandlingArea, AirOperationsArea, Clinic, FireStation, PoliceStation, QuarantineStation, and so forth.
Time	TimeInterval, ExactTime, and so forth.

Table 2: Extension ontology classes for smart airport.

Classes	Subclasses
Event	USNInfo, LocationInfo, TagInfo, BehaviorInfo, AlertInfo, and so forth.
Service	Security, Business, Administration, and so forth.
Vehicle	CargoDolly, CargoLoader, Trailer, Truck, Bus, FuelingVehicle, PassengerStair, RampEquipment, PassengerCart, and so forth.
Cargo	Freight, Baggage, Animal, and so forth.
Aircraft	Airplane, Helicopter, and so forth.
Agent	Airlines, Logistics, Police, Firefighting, Hospital, Construction, Catering, Cleaning, Store, and so forth.
SmartDevice	RFIDReader, USNSensor, CCTV, WirelessAP, SmartPole, and so forth.
MobileDevice	Smartphone, Laptop, HandheldDevice, Camera, and so forth.
ITSystem	AerialControl, CargoHandling, PassengerMonitoring, FacilityManagement, BuildingOperation, ImmigrationControl, SecurityControl, FlightSchedule, AgentITSystem, and so forth.
Pub-Transportation	Rail, GroundBus, Taxi, and so forth.

Table 3: Ontology properties.

Property categories	Properties
Object properties	hasAccess, entersInto, isOwnerOf, isOwnedBy, exitsFrom, checksIn, checksBaggageIn, isOnBoard, isMovingBy, isLoadedOn, isFoundAt, isIdentifiedBy, isAlertFrom, operates, isOperatedBy, isAdministratedBy, and so forth.
Data properties	hasID, hasTimeInterval, hasExactTime, hasLocationInfo, hasSecurityLevel, hasThreatLevel, hasAlertLevel, opensAt, closesAt, entersAt, exitsAt, and so forth.

Once the XOntology has been developed, it can be shared and reused by many applications. Before we proceed to develop the ontology hierarchy of a domain, we need to list the key terms (entities) and their properties as in the previous two tables. After then, the derived terms are defined as classes of the domain and relationships as properties in Figure 6. The instances of classes are defined as individuals associated with the properties of the XOntology (see Figure 7). Individuals are introduced within the reasoning framework as instances of individual classes. They are interrelated with the corresponding properties in the form of RDF triples.

OWL as an ontology language has three types of properties: object properties, data properties, and annotation properties [30]. The object properties explain the relationships between two resources (individuals of classes), whereas the data properties relate a certain data value to an individual. The annotation properties are used to add metadata to classes, properties, and individuals.

3.3.4. Knowledge Reasoning with SPARQL Queries. SPARQL is designed to query RDF [47]. The query language structure is similar to SQL query processing scheme in that it allows the query processor to search for data that meet the conditions. SPARQL queries are used to determine whether a certain activity or behavior is supposed to occur within the ontology-enabled context-aware knowledge framework. SPARQL query lists the latest occurrence of individuals from the RDF triples in Figure 8. In this example, "Yanggon_Kim" is not supposed to be found at the security ID area. By combining this result and situation analysis rule, the framework performs the contextual reasoning and shares the result with other correspondent parties.

By allowing SPARQL queries to find logical anomalies against the predefined properties, we can keep the whole framework in place. Further analysis based on the CAI model is discussed in the following section.

4. CAI-Based Situation Analysis (SA)

The CAI model is supported by the object properties and the data properties which are used as assumptions and constraints for the fundamental situation analysis and categorized in the form of the SA rule from the development of the previous context-aware collaboration framework [5–7]. The SA rule is composed of the predefined behavioral norms including existence availability (EA); the access availability

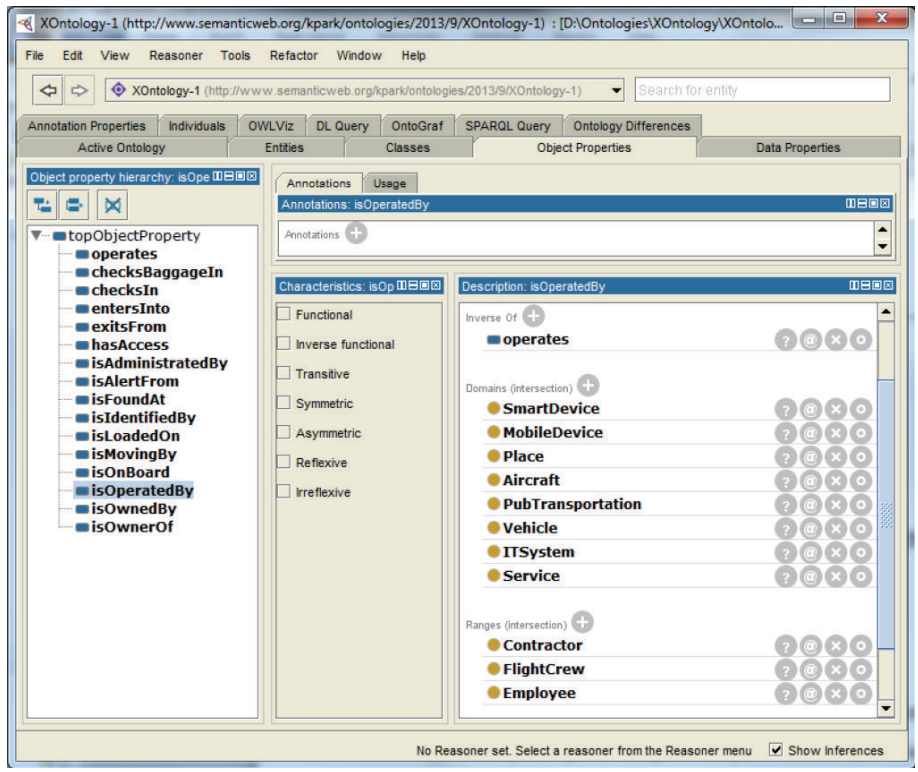

Figure 6: XOntology properties.

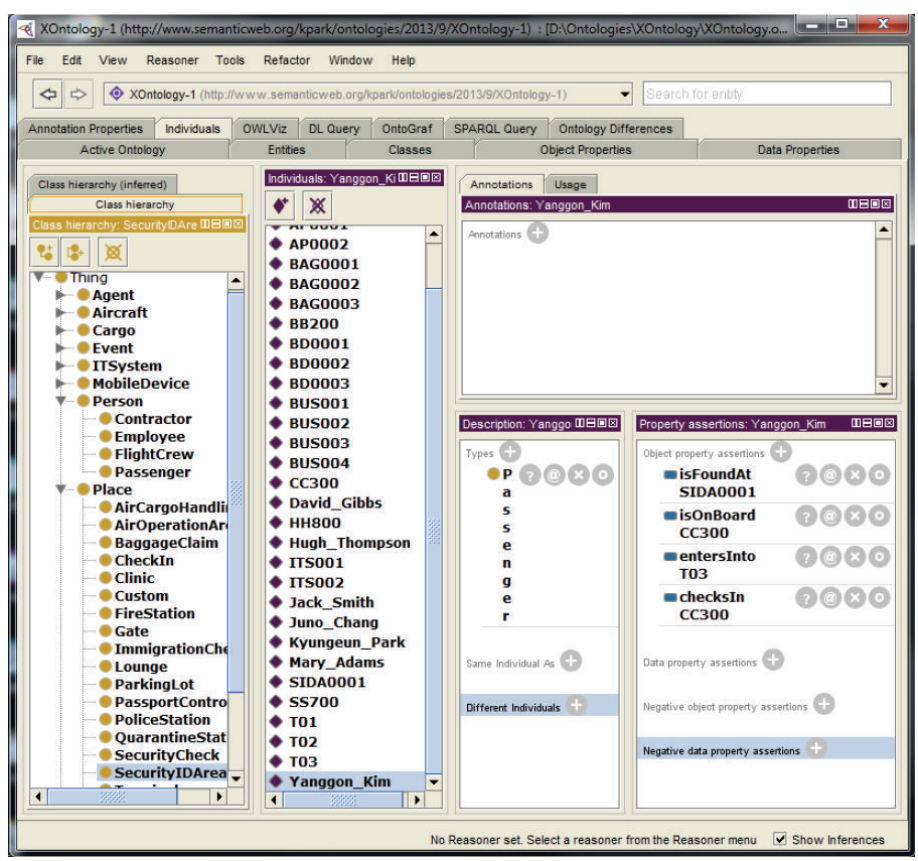

Figure 7: Ontological relationship.

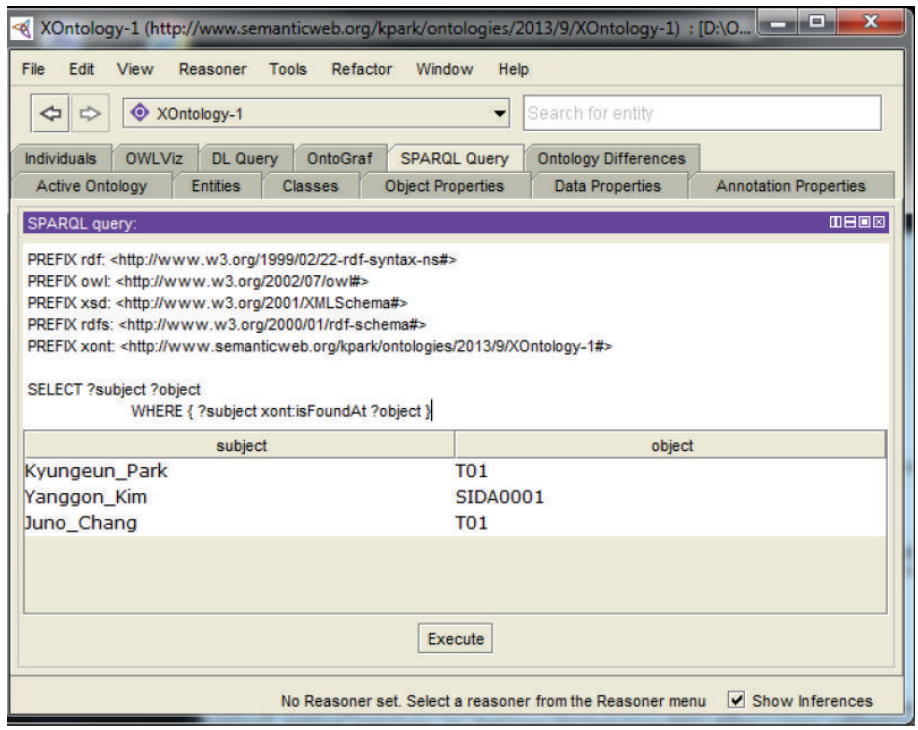

Figure 8: SPARQL queries for situation analysis.

```
EA Violation::Kyungeun Park: @SecurityCheck and
@ImmigrationCheck
```

Algorithm 1: Matching result of EA norm.

```
AA Violation::David Texas (C): appeared @
AirOperationsArea illegally.

AA Violation::Amanda Maryland (C): appeared @
AirOperationsArea illegally.

AA Violation::Kyungeun Park (T): appeared @
ImmigrationCheck illegally.
```

Algorithm 2: Matching result of AA norm.

(AA); the expected transit time (ETT); and the sensor range (SR). All the facts like real-time contextual information such as tag or location information, timestamp, USN sensors data, and any other behavioral activities are reviewed to decide if they are eligible for the semantic guidelines of the SA rule.

4.1. Existence Availability (EA) Norm. The expected pairs of places and their associated availability of simultaneous existence are used for validating identification results and figuring out the abnormal identification. Based on the EA norm, the ontology context reasoned is to decide whether a person could appear at different places at the same time.

Algorithm 1 shows that a person ("Kyungeun_Park") is identified at different places at the same time and indicates that she may cause a certain security violation.

4.2. Access Availability (AA) Norm. The AA norm is used for validating whether any person is passing or staying in unapproved area, scanning all recognized person objects in working memory for any possibility of illegal pass. The AA norm validates whether identified person or employees at a certain place of an airport are eligible to be there with a proper ID card or boarding pass.

Algorithm 2 shows several access violations resulted from illegal access with invalid ID. For example, in order to access

```
ETT Violation::Amanda Maryland: 75(min) spent
between @ Lounge and @ Gate020
```

Algorithm 3: Matching result of ETT norm.

```
SR Warning::Ozon @ AirCargoHandlingArea: 0.15
(min) 0.0 (max) 0.12

SR Warning::Lead @ AirCargoHandlingArea: 200.0
(min) 0.0 (max) 100.0

SR Warning::CO @ AirCargoHandlingArea: 12.0
(min) 0.0 (max) 9.0

SR Warning::Noise @ AirCargoHandlingArea: 200.0
(min) 0.0 (max) 80.0
```

Algorithm 4: Matching result of SR norm.

"AirOperationsArea," the "A-" or "B-"type IDs are required and a person with departure boarding pass should not be recognized in "ImmigrationCheck." In addition, departure boarding passes or employee IDs are required to access boarding processing places like "SecurityCheck" or "PassportControl."

4.3. Expected Transit Time (ETT) Norm. The expected transit time rule is used to determine the earliest (MinTime) or the longest (MaxTime) expected time of arrival to move from one location to another. Each combination of two different places, where RFID reader or any other identification facilities are installed, is assigned MinTime and MaxTime, presenting time constraints. This information is used to deduce an abnormal behavior of individual passengers or employees in an airport by comparing time difference between the expected transit time and the actual time spent to move.

Algorithm 3 tells "Amanda Maryland" spent 75 minutes to move from "Lounge" to "Gate020," which takes 15 minutes longer than the maximum transit time, registered with 60 minutes. This matching result will alarm the system administrator and make him (her) keep track of the person and take security measures, if needed.

4.4. Sensor Range (SR) Norm. The allowable ranges of sensors are provided in the XOntology as data properties. An abnormal sensor value is to be identified by the analytic matching procedure of the semantic reasoning agent and the associated action is performed.

In Algorithm 4, abnormal cases are listed after matching the SR norm.

5. Conclusion and Future Work

This research was motivated by matured sensor-enabled smart city environment. Regarding this, we first noticed the enormous amount of sensor events from a variety of city-wide sensor networks. As an effective scheme to handle huge amounts of sensor events, the well-known Hadoop cluster framework was first considered. Mainly, the framework should continuously monitor the whole incoming sensor events and immediately detect a certain abnormal situation from them. Furthermore, the framework needs to work as an integrated collaboration infrastructure for various sensor-enabled application services. As the solution to this, we extended our existing context-aware collaboration framework to be seamlessly integrated with Hadoop cluster framework. This approach encouraged us to efficiently process massive sensor events and semantically analyze the big data within cluster environment. The proposed smart city sensor cloud framework includes Hadoop Distributed File System (HDFS) as the main big data handling scheme of the sensor cloud infrastructure and the context-aware collaboration manager as the semantic analysis infrastructure. The customized MapReduce functions are introduced on top of Hadoop cluster framework to take full advantage of the spatio-temporal locality. The ontology-enabled semantic reasoning scheme with the XOntology is combined with the Context-Aware Inference (CAI) model for various smart city sensor applications. By applying the ontology technology, the proposed framework enhances the availability and interoperability of the contextual information across many cooperating parties according to semantic reasoning results. This framework is flexible enough to include any heterogeneous platforms including many existing IT solutions as well as mobile platforms. Also, this approach is expected to be applied to many existing frameworks where massive contextual data from numerous sensor devices need to be handled in the future smart city environment. As a future work, big data clustering and optimizing schemes and the associated analysis/mining technologies need to be combined with the proposed framework.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] Center of Regional Science, Vienna UT, *Smart Cities—Ranking of European Medium-Sized Cities Final Report*, 2007, http://www.smart-cities.eu/download/smart_cities_final_report.pdf.

[2] T. White, *Hadoop: The Definitive Guide*, O'Reilly Media, Sebastopol, Calif, USA, 3rd edition, 2012.

[3] Apache Hadoop, *Apache Hadoop 2.2.0*, http://hadoop.apache.org/docs/current/index.html.

[4] K. Park, J. Yun, C. Byun, Y. Kim, and J. Chang, "The XCREAM framework and collaboration validity tests," in *Proceedings of the 1st ACIS/JNU International Conference on Computers, Networks, Systems, and Industrial Engineering (CNSI '11)*, pp. 81–88, Jeju, Republic of Korea, May 2011.

[5] K. Park, Y. Kim, and J. Chang, "Interoperable context sharing in an ontology-enabled collaboration framework," in *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication (ICUIMC '14)*, ACM IMCOM, Siem Reap, Cambodia, January 2014.

[6] K. Park, *Collaboration framework for a context-aware environment with multi-device enabled applications [Ph.D. thesis]*, Towson University, Towson, Md, USA, 2012.

[7] K. Park, C. Byun, J. Yun, Y. Kim, and J. Chang, "Context-aware inference (CAI) model on smart computing environment," in *Proceedings of the International Conference on Information Science and Applications (ICISA '12)*, Suwon, Republic of Korea, May 2012.

[8] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop distributed file system," in *Proceedings of the IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST '10)*, Incline Village, Nev, USA, May 2010.

[9] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments," in *Proceedings of the 8th USENIX Symposium on Operating Systems Design and Implementation (OSDI '08)*, 2008, https://www.usenix.org/legacy/event/osdi08/tech/full_papers/zaharia/zaharia_html/.

[10] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," in *Proceedings of the 6th Conference on Operating Systems Design & Implementation Symposium*, vol. 6, Berkeley, Calif, USA, 2004.

[11] S. Khalil, S. A. Salem, S. Nassar, and E. M. Saad, "Mapreduce performance in heterogeneous environments: a review," *International Journal of Scientific & Engineering Research* , vol. 4, no. 4, pp. 410–416, 2013.

[12] B. N. Schilit and M. M. Theimer, "Disseminating active map information to mobile hosts," *IEEE Network*, vol. 8, no. 5, pp. 22–32, 1994.

[13] N. Ryan, J. Pascoe, and D. Morse, "Enhanced reality fieldwork: the context-aware archaeological assistant," in *Proceedings of the 25th Anniversary Computer Applications in Archaeology*, 1997, http://www.caaconference.org/.

[14] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," in *Proceedings of the Workshop on the What, Who, Where, When and How of Context-Awareness*, ACM Press, New York, NY, USA, 2006.

[15] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 2, no. 4, pp. 263–277, 2007.

[16] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *ACM SIGMOD Record*, vol. 33, no. 4, pp. 65–70, 2004.

[17] W3C Recommendation, *RDF Primer*, 2004, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/.

[18] W3C Recommendation, *RDF Semantics*, 2004, http://www.w3.org/TR/2004/REC-rdf-mt-20040210/.

[19] W3C Recommendation, *RDF Vocabulary Description Language 1.0: RDF Schema*, 2004, http://www.w3.org/TR/rdf-schema/.

[20] G. Antoniou and F. V. Harmelen, "Web ontology language: OWL," in *Handbook on Ontologies*, 2004, http://www.cs.vu.nl/~frankh/postscript/OntoHandbook03OWL.pdf.

[21] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing?" *International Journal of Human-Computer Studies*, vol. 43, no. 5-6, pp. 907–928, 1995.

[22] T. Gruber, "A translation approach to portable ontology specifications," Tech. Rep. KSL 92-71, Knowledge Systems Laboratory, Stanford University, 1993, http://www.dbis.informatik.huberlin.de/dbisold/lehre/WS0203/SemWeb/lit/KSL-92-17.pdf.

[23] W3C Recommendation, *OWL 2 Web Ontology Language Primer*, 2009, http://www.w3.org/TR/owl2-primer/.

[24] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American*, vol. 284, no. 5, pp. 34–43, 2001.

[25] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist*, Morgan Kaufmann, Waltham, Mass, USA, 2011.

[26] W3C School, *Introduction to OWL*, http://www.w3schools.com/webservices/ws_rdf_owl.asp.

[27] N. F. Noy and S. W. Tu, "Developing medical informatics ontologies with Protégé," in *Proceedings of the American Medical Informatics Association Annual Symposium (AMIA '03)*, 2003.

[28] P. Borst, "Engineering ontologies," *International Journal of Human-Computer Studies*, vol. 46, no. 2-3, pp. 365–406, 1997.

[29] R. Neches, R. Fikes, T. Finin et al., "Enabling technology for knowledge sharing," *AI Magazine*, vol. 12, no. 3, pp. 36–56, 1991.

[30] N. Drummond, M. Horridge, and H. Knublauch, "Protégé-OWL tutorial," in *Proceedings of the 8th International Protégé Conference*, Madrid, Spain, July 2005, http://protege.stanford.edu/conference/2005/slides/T2_OWLTutorialI_Drummond_final.pdf.

[31] M. Obitko, *Introduction to ontologies and semantic web, extracted from Marek Obitko (advisor Vladimir Marik): translations between ontologies in multi-agent systems [Ph.D. thesis]*, Faculty of Electrical Engineering, Czech Technical University in Prague, 2007.

[32] V. Spiliopoulos, G. A. Vouros, and V. Karkaletsis, "On the discovery of subsumption relations for the alignment of ontologies," *Journal of Web Semantics*, vol. 8, no. 1, pp. 69–88, 2010.

[33] Protégé Team (Stanford Center for Biomedical Informatics Research), *Protégé*, http://protege.stanford.edu/.

[34] M. R. Genesereth and R. E. Fikes, "Knowledge interchange format, version 3.0 reference manual," Tech. Rep. Logic-92-1, Computer Science Department, Stanford University, 1992.

[35] J. F. Sowa, "Conceptual graphs," in *Handbook of Knowledge Representation*, pp. 213–237, http://www.jfsowa.com/cg/cg_hbook.pdf.

[36] Conceptual Graphs, *A World of Conceptual Graphs*, http://conceptualgraphs.org/.

[37] F. Baader and W. Nutt, "Basic description logics," in *The Description Logic Handbook: Theory, Implementation and Applications*, F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., chapter 2, pp. 47–100, Cambridge University Press, New York, NY, USA, 2003, http://www.inf.unibz.it/~franconi/dl/course/dlhb/dlhb-02.pdf.

[38] M. Wolska and M. Regneri, "Description logic in a nutshell," in *Seminar, Resources for Computational Linguists, SS 2007*, 2007, http://www.cse.iitd.ernet.in/~kkb/DL-1.pdf.

[39] F. Baader, I. Horricks, and U. Sattler, "Description logics," in *Handbook of Knowledge Representation*, F. V. Harmelen, V. Lifschitz, and B. Porter, Eds., chapter 3, pp. 135–180, Elsevier, New York, NY, USA, 2008, http://www.cs.ox.ac.uk/ian.horrocks/Publications/download/2007/BaHS07a.pdf.

[40] I. Horrocks and U. Sattler, "Description logics tutorial," in *Proceedings of the ECAI-2002*, Lyon, France, July 2002, http://www.cs.man.ac.uk/~horrocks/Slides/ecai-handout.pdf.

[41] P. McCarthy, "Search RDF data with SPARQL," in *DeveloperWorks of IBM*, 2005, http://www.ibm.com/developerworks/xml/library/j-sparql/.

[42] Apache Jena, *SPARQL Tutorial*, http://jena.apache.org/tutorials/sparql.html.

[43] The Port Authority of New York and New Jersey, *Airport Rules and Regulations*, 2009, http://www.panynj.gov/airports/pdf/Rules_Regs_Revision_8_04_09.pdf.

[44] Siemens, *Siemens Airport—Integrated Solutions*, http://w3.siemens.com/market-specific/global/en/airports/integrated_it_solutions/Documents/Integrated_Airport_Solutions_brochure pdf.

[45] National Institute of Standards and Technology (NIST), *Special Publication 800-12: An Introduction to Computer Security: The NIST Handbook*, 1995, http://csrc.nist.gov/publications/nistpubs/800-12/800-12-html/index.html.

[46] Protégé Team, *User Documentation*, http://protege.stanford.edu/doc/users.html.

[47] B. DuCharme, *Learning SPARQL*, O'Reilly Media, Sebastopol, Calif, USA, 2011.