Hindawi
Journal of Sensors
Volume 2017, Article ID 8790198, 26 pages
https://doi.org/10.1155/2017/8790198

*Research Article*

# SmartOntoSensor: Ontology for Semantic Interpretation of Smartphone Sensors Data for Context-Aware Applications

**Shaukat Ali, Shah Khusro, Irfan Ullah, Akif Khan, and Inayat Khan**

*Department of Computer Science, University of Peshawar, Peshawar 25120, Pakistan*

Correspondence should be addressed to Shah Khusro; khusro@upesh.edu.pk

Received 11 April 2016; Revised 10 August 2016; Accepted 11 January 2017; Published 20 February 2017

Academic Editor: Andrea Cusano

The integration of cheap and powerful sensors in smartphones has enabled the emergence of several context-aware applications and frameworks. However, the available smartphone context-aware frameworks are static because of using relational data models having predefined usage of sensory data. Importantly, the frameworks lack the soft integration of new data types and relationships that appear with the emergence of new smartphone sensors. Furthermore, sensors generate huge data that intensifies the problem of too much data and not enough knowledge. Smarting of smartphone sensory data is essential for advanced analytical processing, integration, inferencing, and interpretation by context-aware applications. In order to achieve this goal, novel smartphone sensors ontology is required for semantic modeling of smartphones and sensory data, which is the main contribution of this paper. This paper presents SmartOntoSensor, a lightweight mid-level ontology that has been developed using NeOn methodology and Content Ontology Design pattern. The ontology describes smartphone and sensors from different aspects including platforms, deployments, measurement capabilities and properties, observations, data fusion, and context modeling. SmartOntoSensor has been developed using Protégé and evaluated using OntoQA, SPARQL, and experimental study. The ontology is also tested by integrating into ModeChanger application that leverages SmartOntoSensor for automatic changing of smartphone modes according to the varying contexts. We have obtained promising results that advocate for the improved ontological design and applications of SmartOntoSensor.

## 1. Introduction

Smartphones are modern high-end mobile phones combining the features of pocket sized communication devices (i.e., mobile phones) with PC like capabilities (i.e., PDAs). Sensory technology was extended to smartphones in order to turn these communication devices into life-centric sensors so that their capabilities and functionalities could be increased substantially. To date, smartphones have a rich set of sensors [1], which have increased their capabilities in several ways especially introducing a new class of cooperative services including real-time healthcare, environmental and transportation monitoring, gaming, safety, and social networking [2, 3]. However, in frame of the smartphone context-aware application development, the integration of large-scale sensory data that contains real-time spatial, temporal, and thematic information, which could be used for decision making in a rich tactical environment, is crucially difficult [3]. Today, most of these context-aware applications use brute-force approach for collecting and analyzing sensory data that wastes valuable energy and computation resources because of generating observations of minimal use.

The huge sensory data obtained from smartphone sensors intensifies the problem of too much data but not enough knowledge, which is undesirable to smartphone context-aware applications. The problem results due to several reasons. First, the heterogeneous sensors produce voluminous data in varying formats and measurement procedures, which makes it difficult for classical Information Retrieval (IR) techniques to help users in searching and retrieving relevant information. Second, sensors differ in their values as well as description terminologies resulting into terminology mismatch, which makes it difficult for keyword-based searching techniques to retrieve relevant information. Third, sensors inherent design characteristics and lack in adaptability to varying conditions hampers the accuracy and reliability of the captured data. Fourth, the noisy, asynchronous, varying sampling rates characteristics of heterogeneous sensors could result into missing of valuable data that limits the capabilities of applications to statically predefined usages of the collected data instead of showing dynamic behaviors. Fifth, the lack and improper definitions of domain and sensors specifications and annotations data can hamper inferencing of domain knowledge from low-level sensory data. Finally, sensors data fusion could enable extraction of knowledge that cannot be perceived or inferred using individual sensors. Therefore, in the light of these problems, the available smartphone context-aware applications and frameworks are ill-equipped at handling raw sensory data, where usage of sensory information is static, predefined, and with no soft integration of new data types from the newly emerging sensors. A slight change in the technologies and conditions would compel for redesigning of an entire application. Furthermore, to develop practically useful application, developers require actionable knowledge, which is not possible from raw sensory measurement information [4]. Therefore, like other real-world sensors-based systems, the data processing, management, and interpretation of smartphone sensory data are a big challenge that can be resolved by either smarting applications or data. The later approach is more practical by leveraging state-of-the-art technologies for more meaningful representation and semantic interpretation of smartphone sensors and sensors observations for using in potential context-aware applications.

The potential of sensor technology cannot be optimally exploited until the availability of a well-developed common language for expressing different aspects of sensors [5]. Sensors and sensors observations have already been standardized for improving interoperability among heterogeneous sensors data repositories and applications [6]. These standards, however, provide synthetic interoperability with no facilities of semantic descriptions for computer logic and reasoning [7]. Therefore, Semantic Web technologies could be used to provide semantic layer to enhance semantic compatibility and interoperability of smartphone sensors and data. In this regard, ontologies allow the annotation of sensory data with spatial, temporal, and thematic metadata to enhance semantic understanding, interoperability, and mapping of relationships between mismatching terms to improve performance of a system [8]. Therefore, smartphone sensors ontology is immensely required to provide a common and widely accepted language as well as dictionary of terminologies for understanding the structure of information regarding smartphones, sensors, and sensory observations in order to provide highly expressive representations, advanced access, reuse smartphone and sensors domain knowledge, formal analysis of sensors resources and data, and mapping high-level contexts and to make explicit the domain knowledge without having knowledge of technical details regarding format and integration. Furthermore, the ontology would allow for classification of the capabilities and observations of sensors, provenance of measurements, reasoning about individual sensors, connecting a number of sensors as a microinstrument, semantic interoperability and integration, and other types of assurance and automation. Annotating sensory data would enhance sensors data fusion and interoperability between heterogeneous sensors, reused as compared to syntactic representation, and contextual information for situation awareness. The ontology would revolutionize smartphone context-aware applications by providing a broader data model with the potential for integrating new and emerging contents and data types. The ontology would separate the application knowledge from the operational knowledge, which would enable application and knowledge management easier and bring semantic interoperability among applications [9].

The objective of this paper is to design and develop an ontology for smartphone sensors, namely, SmartOntoSensor (SOS) that consists of formal conceptualization of smartphone in general and smartphone sensors in specific for context representation. SmartOntoSensor has numerous characteristics including (1) semantic annotation of smartphone and sensors to increase data interoperability, (2) fusion of multisensors data to support intelligent decision making, (3) resolution of sensors data heterogeneity to express data uniformly, (4) description of sensing and measurements capabilities of sensors to increase sensors data sharing and reusing capabilities, and (5) addition of contexts and contextual information to support context-aware applications. The SmartOntoSensor framework is kept conservative as much as possible while keeping the option open for changes, potential reuse, extension, and plugging into other smartphone domain-specific heavy-weight ontologies. This is due to the speedy evolution in the smartphone's hardware and software industries, which makes it essential that decisions made today regarding smartphones and associated sensors specifications are adoptable and extensible. This paper presents a pragmatic approach for the development of smartphone sensors ontology with no claim that orthogonal, complete, or universally acceptable smartphone sensors ontology is feasible. However, SmartOntoSensor is evaluated using state-of-the-art technologies and standards, and the results are promising. The rest of the paper is organized as follows: Section 2 briefly presents related work, Section 3 presents the proposed SmartOntoSensor ontology, Section 4 presents evaluation and discussion, Section 5 presents some of the potential applications of SmartOntoSensor, and finally Section 6 concludes our discussion. References are presented at the end.

## 2. Related Work

Sensor networks empower Internet with the acquisition of contextual information by observing and measuring real-world incidents and pave the way for the creation of context-aware platforms, applications, and services. However, to gain high degree success and adoptability, data captured by different types and levels of sensors in sensor networks need to be utilized productively. Despite the extensive research efforts in sensors and sensor networks technologies, a universally accepted language for representing sensors' definitions, properties, taxonomies, performance descriptions, and so forth was needed to enhance data fusion and interoperability in a network-centric environment [10]. Therefore, several researchers have investigated ontologies for semantic representation of sensors and sensor networks and came up with a number of ontologies and ontological models [4, 5, 7, 8, 10–15], which are briefly described here.

Avancha et al. [11] used ontology for adaptive wireless sensor networks enabling sensor nodes adoptable to available power, environmental factors, and current operating conditions while maintaining calibration and communication. Its usefulness is the concepts representing a sensor node as a system having different components and their relationships. OntoSensor [5, 10] is a general knowledge base of sensors for query and inference and adapted its concepts from SensorML [16], IEEE SUMO [17], and ISO 19115. However, it lacks distinctive data description model to facilitate interoperable data representation for sensors observation and measurement data. In addition, it provides no constructs to describe sensors as process similar to SensorML [7]. Eid et al. [13] have described a two-layer high-level framework for universal sensors ontology in order to describe hierarchical knowledge model of sensors, dynamic observational properties of transducers, and the integration of domain-specific ontologies with the ontology. Important aspects of the ontology are using notion of virtual sensors formed by physical sensors for more abstract measurements and operations. However, the ontology mainly focuses on data and measurements with little capacity to describe sensors, systems, and how measurements are taken. CSIRO [7] is a generic ontology that organizes concepts into four core clusters while covering a broad range of concepts for describing sensors, groundings, operational model, process, and measurement. However, the ontology contains no concepts for describing components of platforms. Coastal Environmental Sensing Networks (CESN) ontology [12] contains description of sensors types, deployment, location, and physical property. The CESN ontology is, however, very limited of having 10 concept definitions for sensor instances and 6 individuals [18]. Semantic Sensor Network (SSN) ontology [8] has reused DUL (DOLCE Ultra Lite) upper ontology for describing sensors in terms of measurement capabilities, observations, sensing methods, deployments, and operating and survival ranges along with performance within those ranges for enhancing discovering and querying sensors in a network-centric environment. The ontology is more general and comprehensive because of providing most of the necessary details about different aspects of sensors and measurements and can pave the way for the construction of any domain-specific sensors ontology. SWAMO ontology [19] provides interoperability between sensor web products and services. It includes concepts for sensors and actuator and enables autonomous agents for system-wide resource sharing, distributed decision making, and automatic operations. The ISTAR ontology [14] solves the problems faced by intelligence and surveillance including dynamic selection and assignment of sensors (i.e., depending on requirements, fitness, etc.) for individual tasks in a mission. Service-oriented sensor network ontology [15] is developed using Geography Markup Language (GML), SensorML, SUMO, and OntoSensor ontology for describing sensors services. The ontology emphasizes developing sensors descriptions ontology for sensors discovery and description of sensors metadata in a heterogeneous environment. Korpipää and Mäntyjärvi [4] have designed ontology for mapping raw sensory data from mobile devices for high-level semantic description of composite contexts. The ontology encourages the quick development of sensors-based mobile applications, more efficient use of development and computing resources, and reuse as well as sharing of information between communicating entities. However, the ontology provides no description of sensors and platforms and their relationships with contexts. OOSTethys [20] is an observation-centered ontology describing observation as a procedure for estimating property value of a feature of interest and process as a system comprising other systems or atomic processes.

Sensors used in different applications including home appliances, robotics, military, and earth sciences can have analogous characteristics but their needs and usage make them unique [21]. For example, sensors used in weather forecasting measure basic physical properties including air pressure, humidity, temperature, and wind speed [21], whereas sensors used in military missions provide information about hostile terrain such as tactics, location, movement, strength, and equipment and in the development of countertactics and strategies [10]. Smartphone sensors have the same sensing capabilities but with more potential usage than sensor nodes found in sensor networks. This is because, in addition to sensing capabilities, these sensors are locally supported by rich processing, storage, and communication capabilities that are integrated in a single unit and turn smartphones into smart sensors. However, the available ontologies for sensors and sensor networks [4, 5, 7, 8, 10–15] cannot be applied directly for smartphone sensors-based context-aware computing due to a number of shortcomings. These include (1) variations in scope and completeness, where the ontologies were exclusively developed for sensors and sensor networks by describing heterogeneous sensors nodes and enhancing data fusion and interoperability in a network-centric environment, (2) lack of unified ontology framework and consistency in definitions of concepts leading to poor reuse and sharing, and (3) lack of expressiveness due to low explicit hierarchy of concepts and poor logic of relationships which could result into unsatisfactory reasoning [22]. In addition, technically they tend to be shallow and provide superficial aspects of sensors that are expressed in taxonomies and captured as class hierarchies [10]. Therefore, none of the existing sensors and sensor networks ontologies is detailed enough to provide constructs that could be applied directly to meet the unique needs, features, and applications of smartphone and associated sensors in real-world scenarios. However, these ontologies contain important ingredients (i.e., concepts, relationships, and axioms) that can be reused to provide necessary grounds and understanding for smartphone sensors ontology.

## 3. SmartOntoSensor Ontology

The lifestyle of people changes with the developments in the society resulting into new events, interactions, and needs. Smartphones, because of their sensing capabilities, have the potential to capture these aspects and understand the needs of the users. However, due to complexities in understanding users' lifestyles, smartphone sensors are needed to perceive complex objects and their actions as well as interactions effectively under varying operating conditions, strict power constraints, and highly dynamic situations. Furthermore, smartphone-based context awareness works by capturing excessive sensory measurements for inferring complex contextual information including information about environmental conditions, identities of objects in the environment, physical activities of objects and their positions as well as interactions, and the undergoing tasks. Such context-aware smartphone sensors applications demand for comprehensive semantic modeling of smartphone sensors data. It is observed that the role of smartphone sensors ontologies is inevitable for improving the power of smartphone context-aware applications. In order to meet the unique needs and applications, comprehensive smartphone sensors ontology is demanded, which consists of domain theory, represented in a language, and constructed using the functional and relational basis to support ontology-driven inference.

The intended purpose of the SmartOntoSensor is to develop an ontological model consisting of formal conceptualization of smartphone resources and sensors including their categories, taxonomy, relationships, and metadata regarding sensor characteristics, performance, and reliability. In addition, the SmartOntoSensor includes logical statements that describe associations among components and sensor concepts as well as aspects of their operating principles, computing and capabilities, platforms, observations and measurements, and other pertinent semantic contents. The primary objectives of SmartOntoSensor include the following.

(i) Providing a semantic framework for capturing the important functionality features of smartphone sensors enabling context-aware applications to reason the available and running sensors for applying them to current information needs, querying, and retasking as needed.

(ii) Providing context-aware applications with a semantic interface for managing, processing, integrating, and making sense of data acquired from a set of heterogeneous smartphone sensors.

(iii) Providing semantic description of smartphone sensors for reasoning available sensors capabilities and performances to construct low cost combinations of sensors for achieving goals of an operation.

(iv) Providing basis for new measurement methods to evaluate each perception system's ability (sensors and algorithms) to perform the required tasks.

The development of SmartOntoSensor is an attempt to provide a common understating of data captured by smartphone sensors to increase information value and reusability for the application development and sharing. The goals and design principles suggested by [4] are followed in the development of the ontology to ensure its coverage, validity, and usability:

(i) The ontology has been developed for representing information in domain of smartphone and sensors and mapping sensory information into high-level contexts to be used in a variety of context-aware applications.

(ii) The ontology describes concepts, relations, and expressions in a simple and easy-to-understand manner to be easily and effectively used by application developers.

(iii) The ontology is flexible and extendable as it allows developers/users with minimal overhead to add new domain-specific concepts and complementary relations in order to enhance interoperability and knowledge sharing among smartphone context-aware applications.

(iv) The ontological representation facilitates inference by allowing developers to employ an efficient inference method using recognition engines or application control.

(v) The ontology is general by describing concepts, facets, and relationships that are possibly applicable to a wide range of smartphone platforms, embedded sensors, data formats, measurement capabilities and operations, and contexts.

(vi) The ontology is memory-efficient and supports time-efficient inference methods. The imports and constructs in the ontology are defined while keeping in mind the limited nature of memory and processing resources of smartphones; that is, a smartphone should not be jeopardized during inferencing task.

(vii) The ontology provides detailed information about ingredients and the versatility of expressions is high. The ontology is lightweight but comprehensive by declaring enough concepts and relationships to describe possibly every aspect of the domain of interest.

(viii) The concepts and properties in the ontology are defined and arranged precisely for ease in access and producing potentially high values for quality and completeness metrics. In addition, the ontology can be easily used in any of the smartphone Semantic Web frameworks such as AndroJena.

*3.1. Materials and Methodology.* In developing SmartOntoSensor, the NeOn methodology [23] is used, which emphasizes the searching, reusing, reengineering, and merging of ontological and nonontological resources and reusing ontology design patterns, which are the main designing rationales of SmartOntoSensor. However, the NeOn methodology [23] is lacking with ontology project management features, which are adopted from the POEM methodology [24]. Good ontological engineering emphasizes leveraging of upper and related ontologies consisting of general ingredients and providing a common foundation for defining concepts and relationships in a specialized domain-specific ontology. The use of standard ontologies implies shorter

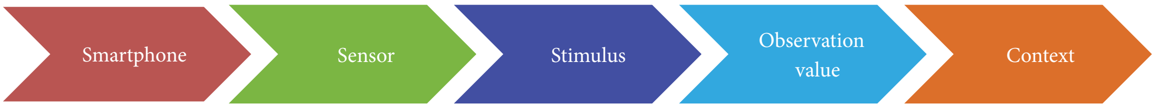


Figure 1: 3SOC ontology design pattern for SmartOntoSensor.

Table 1: An excerpt of the lexicon.

| Lexicon | | | |
|---|---|---|---|
| Smartphone | WiFi | Type | IMEI |
| Accuracy | Resolution | Humidity | Latitude |
| Location | Physical sensor | Service | Country |
| Manufacturer | Accelerometer | Sensing | Roll |
| Bluetooth | Observation value | Logical sensor | Manufacturer email |
| Temperature | Hardware | Profile | Multitouch |
| Version | Passive sensing | Time | Unit |
| Input | Output | Single output | Event status |
| Pitch | Active sensing | Observation | Indoor range |

development cycles, universalism, initial requirements set identification, easier and faster integration with other contents, and more stable and robust knowledge systems [13]. The Content Ontology Design pattern [25] is used where contents in SOS are conceptual and instantiated from logical upper-level ontologies and provide explicit nonlogical vocabulary for the domain of interest. Furthermore, Componency Pattern is used for representing classes/objects as either proper parts of other classes/objects or having proper parts. Technically, the Stimulus-Sensor-Observation pattern [8] is extended into Smartphone-Sensor-Stimulus-ObservationValue-Context (3SOC) design pattern (shown in Figure 1) in order to represent the relationships and flow of information between the components from inception to application. Verbally, a smartphone system contains sensors for detecting stimulus containing observations for producing an observation value, which could be used to identify a context for invoking a service to take an appropriate action.

*3.1.1. SmartOntoSensor Requirements Specifications.* The ontology requirements specification activity is performed for identifying and collecting requirements that the ontology should fulfill. Ontology Requirements Specification Document (ORSD) is formed explaining (1) purpose and reasons to build the ontology; (2) scope of the ontology to fuel applications for mapping smartphone sensory data into high-level contexts for adopting services according to the contexts; (3) users and beneficiaries of the ontology who will be developing applications/services that interact with smartphones and services; (4) ontology as a knowledge base to store data about smartphone, resources/devices, sensors, measurement capabilities and properties, contexts, services, profiles, and so forth; (5) and degree of formality of the ontology by implementing in OWL-DL to get maximum expressiveness with computational completeness.

The SmartOntoSensor requirements are mainly concerned with nonfunctional and functional requirements. The nonfunctional requirements comprise terminological requirements (i.e., collection of terms used in the ontology from the standards that could be used to express competency questions) and the naming convention used for the terms. The terminological requirements of SmartOntoSensor can be broadly divided into several categories: (1) base terms represent the basic classes of entities in the domain of interest, which could be further extended into subclasses; (2) system terms represent components, subcomponents, resources, deployments, metadata, and so forth of a system; (3) sensor terms represent types, characteristics, processes, operations, configuration, metadata, and so forth of sensors; (4) observation terms represent input, output, response model, observation condition, and so forth of observations that are used and produced by sensors; (5) domain terms are used for units of measurement, features selections and calculations, sampling patterns definitions, and so forth; (6) context terms are used for recognizing a context such as location, time, event, activity, and user; and (7) storage terms are used for the storage units used for storing sensors captured observations and other data such as file and folder. A lexicon representing a set of terminologies in the problem domain and applications is identified and collected from application-specific and domain-specific documents. An excerpt of the lexicon is reported in Table 1.

The SmartOntoSensor functional requirements represent the intended tasks and are represented in competency questions, which the ontology should answer by executing SPARQL queries such as “what is a smartphone location?,” “what is the sensing accuracy of a smartphone X sensor?,” “which of a smartphone sensors could be used for recognizing a Y context?,” “what is the accelerometer sensor $x$-axis observation value for sitting context,” “what are the environmental conditions for a sensor to work?,” and “what are the humidity level of an environment?” The set of possible competency questions would help in determining the correctness, completeness, consistency, verifiability, and understandability of requirements. The domain characteristics, which are difficult to express in competency questions, are

Table 2: An excerpt of the competency questions.

| Competency question | Question |
|---|---|
| CQ1 | What is status of a smartphone? |
| CQ2 | Is accelerometer sensor available in a smartphone? |
| CQ3 | What is location of a smartphone? |
| CQ4 | What is the output of accelerometer sensor? |
| CQ5 | Which sensors can recognize a running context? |
| CQ6 | What are the device conditions for GPS sensor to work? |
| CQ7 | Who is user of a smartphone? |
| CQ8 | Which service would be invoked if sleeping context is detected? |
| CQ9 | What are the activities composing a sensing process? |
| CQ10 | Which of the sensors are used by accelerometer as supporting sensors for detecting a running context? |

Table 3: Relationship between scenarios and iterations.

| Scenarios | | Iterations | | |
|---|---|---|---|---|
| | | 1st | 2nd | 3rd |
| 2 | Reusing and reengineering nonontological resources | | √ | |
| 3 | Reusing ontological resources | √ | | |
| 4 | Reusing and reengineering ontological resources | √ | | |
| 5 | Reusing and merging ontological resources | √ | √ | |
| 6 | Reusing, merging, and reengineering ontological resources | √ | √ | |
| 7 | Reusing ontology design patterns (ODPs) | √ | √ | √ |
| 8 | Restructuring ontological resources | | | √ |
| 9 | Localizing ontological resources | | | √ |

written in natural language sentences such as “fusion of data from multiple sensors for mapping a context” and “extreme environmental conditions can affect sensors observations and performances.” The initial iteration produces short list of functional requirements. However, the list is improved significantly in the subsequent iterations and consists of 156 competency questions and 40 domain characteristics. An excerpt of competency questions is shown in Table 2.

*3.1.2. SmartOntoSensor Development Resources and Tools.* By following the NeOn methodology [23], SmartOntoSensor is developed by reusing the existing knowledge resources. The development task is divided into three iterations where both ontological and nonontological resources are reused in the first and second iterations, and ontology design pattern is included in all of the development iterations. Table 3 shows the selected scenarios to be carried out in combination with Scenario 1.

SmartOntoSensor is constructed by reusing multiple relevant ontologies. The review and analysis of the sensors and sensor networks ontologies and sensors vocabularies have highlighted that reusing available sources describing sensors, their capabilities, the systems they are part of, observation and measurements, properties and associations, quantitative values for properties, and so forth can provide promising start for building SmartOntoSensor. After thoroughly analyzing the sensors and sensor network ontologies, the SSN ontology [8] is found most relevant due to providing advanced schema for describing sensor equipment, observation measurement, and sensor processing properties. SSN has a wider range of generality and extension space and is reused in several projects to solve complex problems [26]. Therefore, SSN is extended for the development of SmartOntoSensor. Other ontologies are also found containing relevant ingredients but excessive imports can cause certain problems including decrease in efficiency, simplicity, consistency, verifiability, and flexibility [27]. Some categories, taxonomies and definitions of commonly used concepts, properties, and metadata are adopted in part from SensorML [16]. Although the initial objective was to faithfully replicate the required items from SensorML, some implementation compromises and workarounds are made exclusively to meet the unique demands of the new paradigm. The context ontology (CXT) developed by CoDAMoS project [28] is imported and extended with required domain concepts and relationships for modeling context in SmartOntoSensor. Sensors data are stream requiring indefinite timestamp sequence information for unique representation. Therefore, OWL Time (TIME) ontology is reused for incorporating time information into the SmartOntoSensor. To develop SmartOntoSensor, classes in the imported ontologies are either used directly or extended by making SmartOntoSensor classes as subclasses of the relevant classes. Furthermore, classes in the imported ontologies representing the same concepts are aligned by declaring them equivalent classes and other classes are refactored either by restructuring the class hierarchy or by defining new associations and relationships for enabling them to be used in SmartOntoSensor as per needs. The

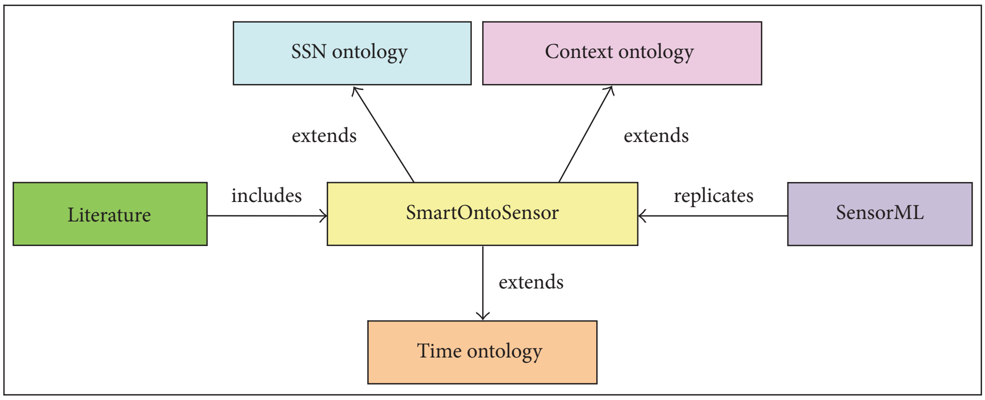


Figure 2: Abstract level structure of SmartOntoSensor.

additional domain-specific contents of SmartOntoSensor are captured from the detailed investigation and analysis of the related literature. Figure 2 shows the abstract level structure of SmartOntoSensor by highlighting the contributing information sources. To communicate semantics of sensors observations, an appropriate terminology (obtained in lexicon) is defined for describing concepts, relations, and processes. The terminology used is discussed in the subsequent sections. After formally defining the constituents, SmartOntoSensor is developed in OWL-DL language using open source ontology editor and knowledge-based framework Protégé 4.3 along with its exclusive plug-ins (e.g., SPARQL query plug-in and RacerPro reasoner) for the ontology editing, development, implementation, and testing.

*3.2. SmartOntoSensor Framework.* The SmartOntoSensor framework is conceptually (but not physically) organized into 9 modular subontologies where a modular subontology represents a subdomain and a central ontology links the other ontologies. Each of the subontologies contains a number of concepts and properties for modeling a specific aspect of the domain of interest. The SSN framework [8] has provided inspiration for the development of SmartOntoSensor framework and some of the its modules are refactored and merged for inclusion into the SmartOntoSensor framework. The novelty of the framework is fourfold information presentations to fulfill objectives of SmartOntoSensor. First, detailed information about smartphone systems regarding their hardware components, software, platforms, metadata, potential deployments, and so forth is presented. Second, the detailed information about smartphone sensors regarding their inputs, outputs, processing, observations, measurements, operating restrictions, capabilities, and so forth is presented. Third, potential applications of observation values (i.e., produced by smartphone sensors after sensing process) for context recognition and context modeling as a whole such as user information and profiles, current and planned activities and events, and device and its surroundings are presented. Fourth, enhancing of smartphone context-awareness capability for solving the challenges of applications adoptability according to users' contexts is presented. Figure 3 depicts the SmartOntoSensor framework and a snippet of the main concepts and their relationships. The framework represents the main high-level concepts and object properties in a subdomain, whereas detailed concepts and object properties are left aside. In the framework, classes are represented with rectangles and subclass axioms and object properties are represented with solid and dotted arrow lines, respectively. The detailed discussion on the SmartOntoSensor framework's subdomains is presented in Sections 3.2.1 to 3.2.8.

*3.2.1. Smartphone.* The smartphone subdomain is constructed around using concepts for modeling knowledge about a smartphone system describing its resources (i.e., hardware and software), organization, deployment, and platform aspects. The SOS:Smartphone concept is derived from the SSN:System concept of the SSN ontology [8] providing necessary properties for deployment and platform. SOS:Smartphone could have different hardware resources including sensing (SOS:SensingResource), memory (SOS:MemoryResource), and network (SOS:NetworkResource) and software resources including operating system (CXT:OperatingSystem) and middleware (CXT:Middleware). Each of these resources is a system by itself. A number of object properties (i.e., SOS:hasBluetooth, SOS:hasWiFi, SOS:hasSensor, hasOperatingSystem, etc.) are made as subproperties of SSN:hasSubSystem for linking SOS:Smartphone with resources. The SOS:Smartphone inherits SSN:hasOperatingRange and SSN:hasSurvivalRange properties to define the extremes of operating environments and other conditions in which a smartphone is intended to survive and operate for providing functionalities including standard configuration, battery lifetime, and system lifetime. A smartphone could be mounted or connected (SSN:onPlatform) with a platform (SSN:Platform ≡ CXT:Platform) which could be having hardware (CXT:providesHardware) and software (CXT:providesSoftware) features. The SOS:Smartphone could be deployed (SSN:hasDeployment → SSN:Deployment) at a specific place including worn on a helmet (SOS:Halmet ⊆ (SSN:Platform ≡ CXT:Platform)) or around the neck, attached with belt (SOS:Belt ⊆ (SSN:Platform ≡ CXT:Platform)), placed on a selfie stick (SOS:SelfiStick ⊆ (SSN:Platform ≡ CXT:Platform)), and placed in treasure pocket. SSN:Deployment could be a complete process of

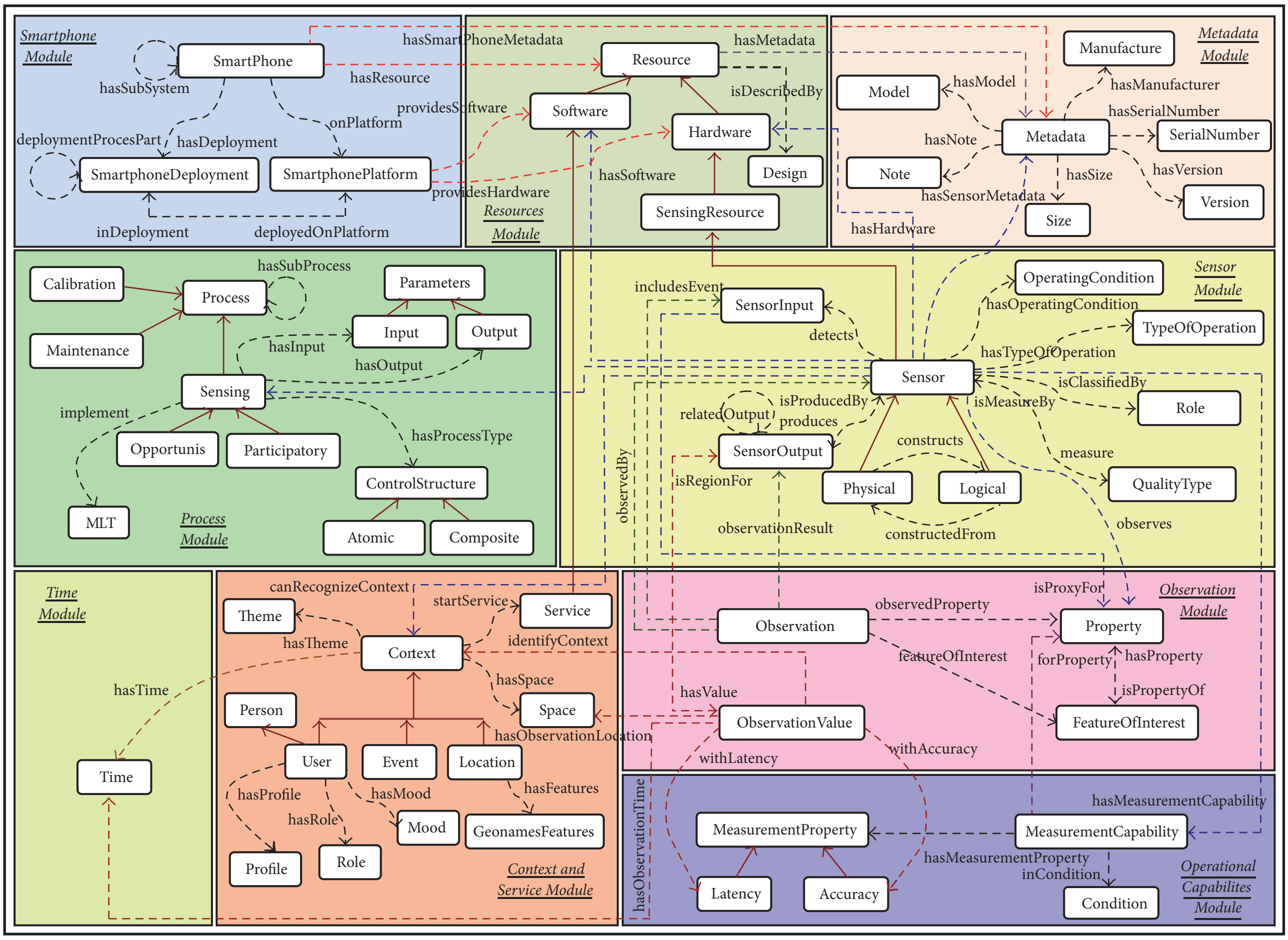


FIGURE 3: SmartOntoSensor framework.

installation, maintenance, and decommission and could have spatial (SOS:hasDeploymentPosition) and temporal (SOS:hasDeploymentTime) properties.

*3.2.2. Sensor.* The sensor subdomain is the cornerstone of SmartOntoSensor that serves as a bridge for connecting all other modules. It models smartphone sensors using concepts and properties for describing taxonomy of sensors, types of operations, operating conditions, resource configurations, measuring phenomena, and so forth. During the alignment process between SSN and SmartOntoSensor, the SSN:Sensor concept is extended by a detailed hierarchy of sensors in SmartOntoSensor. The SSN:Sensor is categorized into logical (SOS:LogicalSensor) and physical (SOS:PhysicalSensor) sensors where physical sensors represent hardware-based sensors and logical sensors represent software-based sensors that are created by employing one or more physical sensors. Individual sensors (e.g., SOS:Accelerometer) are declared subclass of either physical or logical sensors. A physical sensor has type of operation (SOS:hasTypeOfOperation) which could be either active sensing (SOS:ActiveSensing) or passive sensing (SOS:PassiveSensing). A physical sensor has to work under certain conditions and have specific features such as SSN:Accuracy, SSN:Latency, SOS:Hystheresis, and SSN:Sensitivity to capture a particular stimulus. A sensor could have its own hardware (SOS:hasSensorHardware) and software (SOS:hasSensorSoftware) specifications. A sensor can measure (SOS:measure) properties of a phenomenon that can be quantified (i.e., that can be perceived, measured, and calculated) which could be either physical quality (SOS:PhysicalQuality) or logical quality (SOS:LogicalQuality). A sensor detects (SSN:detects) a stimulus (SSN:Stimulus) which is an event in the real world that triggers a sensor and could be the same or different to observe property and serves as a sensor input (SSN:SensorInput). The output (SSN:"Sensor Output") produced by a sensor can be either a single value (SOS:SingleValue) or more than one value (SOS:TupleValue).

*3.2.3. Process.* In addition to physical instrumentation, a sensor has associated functions and processing chains to produce valuable measurements. The sensor concept has to implement (SSN:implements) a sensing process (SSN:Sensing), which could be either participatory (SOS:Participatory) or opportunistic (SOS:Opportunistic). A process can have a subprocess (SOS:subProcess) such that a sensing process could require a prior calibration process. Therefore, concepts for calibration (SOS:Calibartion) and maintenance

(SOS:Maintenance) processes are declared as subclasses of SSN:Process and are disjointed with SSN:Sensing. A sensing process could have input (SOS:Input) and output (SOS:Output) parameters. A sensing process can use calibration and maintenance processes as complementary and supporting classes using SOS:SubProcess property. A process has type (SOS:hasProcessType) either physical (SOS:PhysicalProcess) or logical (SOS:LogicalProcess) where a physical process would be essentially having a physical location or interface and a logical process do not. A sensing process has process composition describing its algorithmic details (i.e., sequence, condition, and repetition) of how outputs are made out of inputs. Therefore, a process has control structure (SOS:hasControlStructure) for linking with instances of SOS:AtomicProcess, SOS:CompositeProcess, and SOS:PhysicalProcess. A sensing process could implement (SOS:implement) machine learning techniques (SOS:MLT) for extracting features of interest from an input to create an output.

*3.2.4. Measurement.* The measurement subdomain is constructed for complementing sensor perspective. The main concepts in this subdomain are SSN:Observation, SSN: FeatureOfInterest, and SSN:ObservationValue. A sensor has an observation (SSN:Observation) representing a situation in which a sensing method (SSN:Sensing) estimates the value of a property (SSN:Property) of a feature of interest (SSN:FeatureOfInterest) where a feature is an abstraction of a real-world phenomenon. An observation is formed from a stimulus (SSN:Stimulus) in a contextual event (SOS: Event) which serves as input to a sensor. A sensor input is a proxy for a property of the feature of interest. An observation (SSN:Observation) should represent an observing property (SSN:observedProperty), sensing method used for observation (SSN:sensingMthodUsed), quality of the observation (SSN:qualityOfObservation), observation result (SSN:observationResult), and the time (CXT:Time) at which the sampling took place (SSN:observationSamplingTime). An observation result is a sensor output (SSN:SensorOutput), which is an information object (SSN:SensorOutput ⊆ SSN: InformationObject). The information object represents a measurement construct for interpreting events, participants, and associated result and signifies the interpretative nature of observing by separating a stimulus event from its potential multiple interpretations. SSN:SensorOuput has value (SSN:hasValue) for observation value (SSN:ObservationValue). The SSN:ObservationValue represents the encoding value of a feature and is indirectly depending on the accuracy, latency, frequency, and resolution of a sensor producing output. The observation value is annotated with location (SOS:hasObservtionLocation), theme (SOS:hasObservationTheme), and time (SOS:hasObservationTime) information for identifying a context (SOS:identifyContext).

*3.2.5. Capabilities and Restrictions.* Another complementing sensor subdomain is the sensor measurement capabilities and operational restrictions. Sensors are integrated within the suit of a smartphone; however, they are intended to be exposed and operated to provide best performance within the defined operating conditions (SOS:OperatingCondition ⊆ SSN:Condition), which are categorized into device conditions (SOS:DeviceCondition) and environmental conditions (CXT:EnviromentalCondition) of the atmosphere (i.e., SOS: Humidity, SOS:Temperature, and SOS:Pressure) in a particular space and time. A sensor has inherent characteristics by design of measurement capabilities (SSN: MeasurementCapabilities) that depends on the measurement properties (i.e., representing a specification of a sensor's measurement capabilities in various operating conditions) and directly affects a sensor's output. The measurement properties (SSN:MeasurementProperty) determine the behavior, performance, and reliability of a sensor. In addition, these measurement properties determine the quality of quantifiable properties such as mass, weight, length, and speed related to a phenomenon. These properties can be classified into accuracy (SSN:Accuracy), frequency (SSN: Frequency), power consumption (SOS:PowerConsumption), random (SOS:Random) and systematic (SOS:Systematic) errors, settling time (SOS:SettlingTime), precision (SSN: Precision), and resolution (SSN:Resolution).

*3.2.6. Metadata.* The metadata subdomain is constructed to provide detailed descriptive information regarding origination, introduction, application, and production of an object or a phenomenon that is of interest to a decision making system. The metadata determines and affects the reliability and credibility of an object or a phenomenon. A vision of SensorML is that the schema should be self-describing and can be accomplished by accommodating metadata about a schema within the schema [10]. An object (e.g., sensor) would have metadata (SOS:hasSensorMetadata) that provides information about manufacturer (SOS:hasManufacturer), model information (SOS:hasModel), serial number (SOS:hasSerialNumber), and size (SOS:hasSize). SOS:Metadata has explicit relationships with SOS:Identification, SOS:Note, and SSN: Design. SOS:Identification provides information about recognition of a phenomenon including manufacturer, model, size, and version. Manufacturer perspective (SOS: Manufacturer) is constructed by providing necessary information about manufacturer of a smartphone, resource, platform, and sensor. SOS:Manufacturer is enriched with several object properties for describing a manufacturer that includes SOS:hasManufacturerEmail, SOS:hasManufacturerName, SOS:hasManufacturerLocation, and SOS: hasManufacturerWebsite. Similarly, additional information about objects are provided by establishing explicit relationships with SOS:Model, SOS:SerialNumber, SOS:Version, and SOS:Size. Similarly, SOS:Note provides a description about a phenomenon.

*3.2.7. Time.* The time subdomain models knowledge about time such as temporal unit and temporal entities. This subdomain has been developed by reusing the OWL Time ontology, in which TIME:CalendarClockDescription, TIME:TemporalUnit, and TIME:TimeZone are made subclasses of CXT:Time. Similarly, day, week, month, second, minute, and hour properties of Time ontology are used to represent time information. SSN:TimeInterval is made subclass of CXT:Time to represent time duration of a phenomenon.

*3.2.8. Contexts and Services.* The context subdomain is constructed for describing the application of sensors' generated observation values for identifying user contexts. Ontology-based context modeling will be helpful in formal and semantic enrichment and representation of complex context knowledge in order to share and integrate contextual information [28]. The context ontology from CoDAMoS project is reused for providing relevant core knowledge for SmartOntoSensor, which is extended with more detailed context types and properties. The SOS: Context is main concept in this subdomain, which is classified into more specialized contexts including CXT:Activity, CXT:Environment, SOS:Event, SOS:Device, ontology:User, and SEN:Location. Each of the subcontexts represents a subdomain which could have more specific contexts such that ontology:Activity is classified into SOS:Motional and SOS:Stationary and SOS:Event is classified into SOS:SpatialEvent, SOS:TemporalEvent, SOS:SpatioTemproalEvent, and so on. The SOS:Device subdomain models knowledge about devices and includes a wide categorization of devices as well as their characteristics. SOS:Environment subdomain models knowledge about environment in terms of environmental conditions such as humidity, noise, light, and temperature. CXT:Location models knowledge about locations such as buildings, location coordinates, spatial entities, distance, and countries. For more detailed location modeling, CXT:Location is linked with geonames: GeonamesFeature. CXT:User (i.e., CXT:User ≡ SSN:Person) subdomain models knowledge about users such as roles, profiles, preferences, tasks, projects, publications, and socialization. For more detailed user mappings, CXT:User is linked with FOAF:Person. SOS:Event subdomain models knowledge about users' real-life events and includes a wide categorization of events using space, time, and other characteristics. CXT:Activity subdomain models knowledge about users' motional and stationary activities and includes their characteristics. A context needs spatial (SOS:Space), temporal (CXT:Time), and thematic (SOS:Theme) information for its description. A context is identified by the observation value (SSN:ObservationValue) that depends on a sensor output (SOS:SensorOutput). CXT:Service subdomain fulfills the service-oriented requirement of the ontology by providing service-oriented features in ubiquitous computing. A service would be software and includes services which would be recognized and utilized on the basis of identified context. A service could have a provider (SOS:Provider) that would include simple or aggregated service providers.

*3.3. SmartOntoSensor Concepts Hierarchy.* SmartOntoSensor taxonomic class diagram, forming foundation of the ontology, is constructed from concepts, which are common and specific to smartphones, sensors, and context applications. The identified concepts are hierarchically arranged by determining their relationships that whether a concept is a subconcept of another concept or not. The required classes, which are provided by either of the imported ontologies (e.g., SSN:Input, SSN:Output, CXT:Software, and SSN:Precision), are directly used in SmartOntoSensor and no explicit classes are declared to eradicate any type of ambiguity. Furthermore, classes in the imported ontologies representing the same semantics are declared as equivalent classes (e.g., SSN:Platform ≡ CXT:Platform). New classes are created explicitly either as parent classes or as subclasses of the relevant classes in the imported ontologies or as per requirements. Figure 4 shows a snippet of SmartOntoSensor concepts hierarchy.

The SmartOntoSensor, at present, contains 259 concepts, where each concept is formed by keeping in mind the unique needs and requirements of the domain. SmartOntoSensor extends SSN ontology by making SOS:Smartphone ⊆ SSN:System, which means SOS:Smartphone is a kind of SSN:System. Smartphone platforms (SOS:Halmet, SOS:Belt, and SOS:SelfiStick), to which a smartphone can be attached during sensing process, have unique characteristics and are made subclass of SSN:Platform ≡ CXT:Platform to partially satisfy needs of a smartphone platform. The SSN:Sensor concept is used to represent sensing devices for SmartOntoSensor to capture inputs and produce outputs. The SSN:Sensor concept is further divided into two categories SOS:PhysicalSensor and SOS:LogicalSensor to represent hardware-based sensors and software-based sensors, respectively, in a smartphone. A SOS:LogicalSensor is formed by employing one or more of the SOS:PhysicalSensor for data capturing and producing a unique output; for example, e-compass sensor is formed using accelerometer and magnetometer sensors. Real-world smartphone physical sensors such as SOS:Accelerometer, SOS:Gyroscope, SOS:Camera, and SOS:GPS are made as subclasses of SOS:PhysicalSensor and logical sensors such as SOS:Magnetometer and SOS:Gravity are made as subclasses of SOS:LogicalSensor. The sensors hierarchy can be extended at any level to include more detailed and specific types of sensors. For example, accelerometer sensor can sense motion in either 2-dimensions or 3-dimensions. Specifically, 1st- and 2nd-generation accelerometer sensors can detect motion in either of these categories while 3rd-generation accelerometer sensors can detect motion across both categories as shown in Figure 5. The quantifiable properties that a sensor can measure (SOS:measure) for a particular phenomenon are classified into physical quality (SOS:PhysicalQuality) or logical quality (SOS:LogicalQuality). A sensor has type of operation, representing how a sensor would sense a stimulus that is either active or passive. Therefore, SOS:ActiveSensing and SOS:PassiveSensing are made as subclasses of SOS:TypeOfOperation. The concepts SOS:DeviceCondition and CXT:EnvironmentalCondition are declared as subclasses of SOS:OperatingCondition. The metadata concepts including SOS:SerialNumber and SOS:Size are declared as subclasses of SOS:Identification, which would be used by SOS:Metadata to provide necessary metadata about smartphones, sensors, and platforms. A smartphone sensor has to perform sensing operations, which could be either opportunistic or participatory. Therefore, SOS:Opportunistic and SOS:Participatory are made as

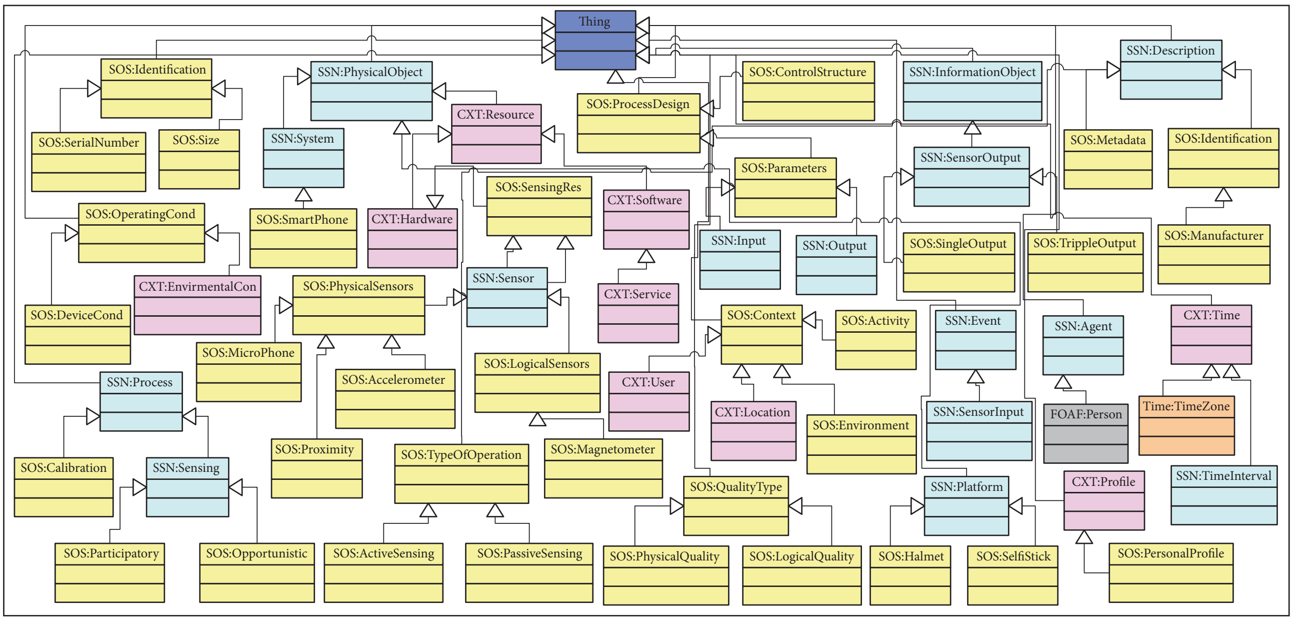


Figure 4: Snippet of SmartOntoSensor concept hierarchy.

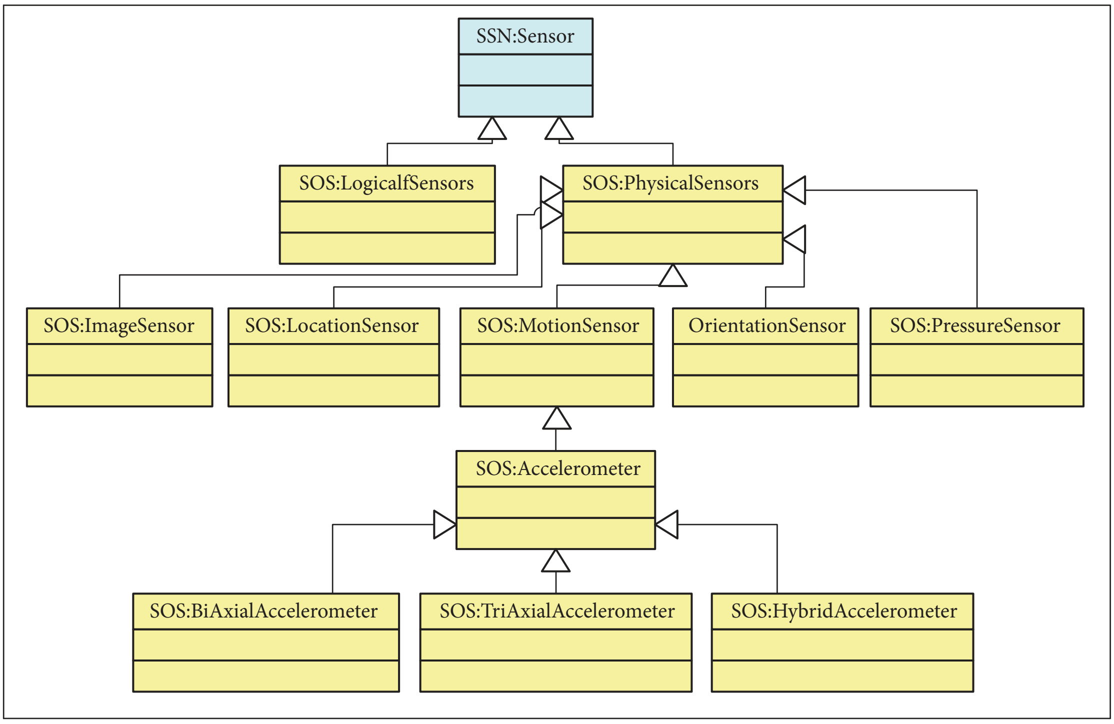


Figure 5: Snippet of SmartOntoSensor “sensor” class hierarchy.

subclasses of SSN:Sensing class. Sensors differ by the amount of output produced where some could produce a single value outputs whereas others could produce triple value outputs. Therefore, SOS:SingleOutput and SOS:TrippleOutput are made as subclasses of SSN:SensorOutput. An observation value produced by a sensor as output could be used to recognize SOS:Context, which is divided into subclasses as SOS:Event, SOS:Device, CXT:User, and CXT:Environment. An identified context can start a CXT:Service, which would be a software and is made subclass of CXT:Software. In addition to all of the above, several concepts are explicitly identified and included in SOS including SOS:RAM, SOS:WiFi, SOS:CPU, SOS:GSM, SOS:Bluetooth, SOS:NFC, SOS:Infrared, SOS:GPU, and SOS:SDCard, as subclasses of CXT:Hardware, SOS:Random, and SOS:Systematic as subclasses of SOS:Error, SOS:Calibration, and SOS:Maintenance as subclasses of SSN:Process, SOS:Theme, and SOS: Space to provide a comprehensive set of information for improving inferencing mechanisms.

Apart from using subclass axioms, classes are coupled with other axioms to facilitate the creation of individuals (i.e., objects) unambiguously and semantically. For example, the disjoint axiom is defined for classes belonging to the same generation level to restrict individuals' behaviors such that SOS:LogicalSensor and SOS:PhysicalSensor are disjointed to ensure that an individual can be an instance of any one of these classes but not both at the same time.

*3.4. SmartOntoSensor Properties and Restrictions.* In OWL, properties represent relationships among classes. A property can be either object property or datatype property, either of which represents link(s) between instance(s) of domain with instance(s) of range. For an object property, both the domain and range should be instances of concepts, whereas for a datatype property domain should be an instance of a concept and range should be an instance of a datatype. Like concepts, exhaustive lists of object and datatype properties are identified from sources for using in SmartOntoSensor.

SmartOntoSensor contains an extended list of object properties (i.e., 382), some of which are coming from the imported sources and others are explicitly created. As several of the SmartOntoSensor concepts are derived from the concepts in the imported ontologies, therefore, their object properties are also inherited in the same fashion and made them more specialized for SmartOntoSensor concepts. The SmartOntoSensor object properties SOS:hasWiFi, SOS: hasBluetooth, SOS:hasCPU, SOS:hasRAM, and SOS:hasDisplay are made as subproperties of the SSN:hasSubSystem for linking SOS:Smartphone with individual resource classes. Another SmartOntoSensor object property SOS: hasSmartDeployment is made as subproperty of SSN: hasDeployment to define relationship between SOS: Smartphone and SOS:SmartPhoneDeployment. It is due to the fact that a smartphone deployment has unique features and methods compared to the objects deployments in wireless sensor networks. As several concepts of the imported ontologies are used directly, therefore, their object properties are used similarly to avoid any confusion. The object properties SSN:hasInput and SSN:hasOutPut are used to represent input and output, respectively, of SSN:Sensing. Similarly, the object property SSN:hasValue is used to represent the relationship between SSN:SensorOuput and SSN:ObservationValue. Furthermore, like concepts, a bundle of domain-specific object properties are identified for SmartOntoSensor from the additional sources to relate concepts in the ontology in a more meaningful and subtle ways such as SOS:recognizeCotnext, SOS:hasProcessType, SOS:organizedBy, SOS:deriveFrom, SOS:constructedFrom, SOS:hasLatency, SOS:hasFrequency, and SOS:hasSpace. Table 4 represents an excerpt of the SmartOntoSensor object properties along with their domains and ranges.

Apart from the object properties, SmartOntoSensor contains an extensive list of 136 datatype properties, which are identified and mapped to give comprehensive information about concepts such as SOS:hasMemorySize, SOS:hasActiveSensing, SOS:modelScientificName, SOS:latitude, SOS:eventCancelled, SOS:isConsumable, SOS:manufacturerEmail, SOS:hasBatteryModel, SOS:minValue, SOS: value, SOS:hasMaxRFPower, and SOS:nickName. Table 5 presents an excerpt of the SOS datatype properties along with their domains and ranges.

Classes in SmartOntoSensor are refined by using object properties and datatype properties to superimpose constraints and axioms for describing their individuals. An example of such constraints is the property restrictions (i.e., quantifier restrictions, cardinality restrictions, and hasValue restrictions) for describing number of occurrences and values of a property essential for an individual to be an instance of a class. For example, for an individual to be an instance of SOS:Smartphone class, it is essential for the individual to have at least one occurrence of SOS:hasCPU object property for relating the individual with an instance of the SOS:CPU class. Tables 4 and 5 also show excerpts of the property restrictions for the SmartOntoSensor object properties and datatype properties, respectively.

## 4. Evaluation and Discussion

Several approaches have been proposed and used for evaluating ontologies from the perspectives of their quality, correctness, and potential utility in applications. The four main methods are gold-standard comparison, application-based evaluation, data sources comparison, and human-centric evaluation [29–31]. The gold-standard method advocates on comparison measurements of a well-formed dataset produced by a given ontology against other datasets. The application-based evaluation method evaluates an ontology using the outcomes of an application that is employing the ontology. The data sources comparison method describes the use of a repository of documents in a domain where ontology is expected to cover the domain and its associated knowledge. The human-centric evaluation method emphasizes human efforts where an expert would assess the quality of a given ontology by comparing it to a defined set of criteria. Similarly, several metrics have been defined for verifying and validating ontologies, where verification determines whether ontology is built correctly and validation is concerned with building the correct ontology [32]. A detailed discussion about ontology evaluation methods, metrics, and approaches is beyond the scope of this paper; however, it can be found in [31]. For evaluating SmartOntoSensor, we have used the first, second, and fourth methods along with comparison using multimetrics approach, accuracy checking, and consistency checking, which are discussed in detail in the subsequent sections. A top-level terminological requirements fulfillment (by providing concepts) comparison of SmartOntoSensor and other sensors and sensor networks ontologies is shown in Table 6. A tick in the table indicates the capability of ontology to describe the stated aspect in some form and absence of a tick indicates either the absence or insufficient information of the aspects.

Table 4: An excerpt of SmartOntoSensor object properties.

| Domain | Object property | Range | Quantifier | Cardinality |
|---|---|---|---|---|
| SOS:Smartphone | SOS:hasCPU | SOS:CPU | Exis. & univ. | Min 1 |
| SOS:Smartphone | SOS:hasOperatingSystem | SOS:OperatingSystem | Existential | Min 1 |
| SOS:Smartphone | SOS:isPlacedOn | SOS:Halmet \| SOS:SelfiStick | Universal | Max 1 |
| SSN:Sensor | SOS:measure | SOS:QualityType | Exis. & univ. | Min 1 |
| SSN:Sensor | SOS:produces | SSN:"Sensor Output" | Existential | Exactly 1 |
| SOS:PhysicalSensor | SOS:hasTypeOfOperation | SOS:TypeOfOperation | Existential | Exactly 1 |
| SOS:PhysicalSensor | SOS:calibration | SOS:calibration | Universal | Nil |
| SOS:LogicalSensor | SOS:constructedFrom | SOS:PhysicalSensor | Exis. & univ. | Min 1 |
| SSN:"Sensor Output" | SOS:relatedOutput | SSN:"Sensor Output" | Universal | Nil |
| SSN:"Sensor Output" | SOS:withAccuracy | SSN:Accuracy | Universal | Nil |
| SOS:Sensor | SOS:hasObservation | SSN:Observation | Universal | Min 1 |
| SOS:QuatityValue | SOS:hasUnit | SSN:"Unit of Mesasure" | Existential | Exactly 1 |
| SOS:Metadata | SOS:hasManufacturer | SOS:Manufacturer | Universal | Min 1 |
| SOS:Metadata | SOS:hasVersion | SOS:Version | Universal | Max 1 |
| SSN:Sensor | SOS:hasEnvironmentCondition | CXT:EnvironmentalCondition | Universal | Nil |
| SSN:Process | SOS:hasProcessType | SOS:ProcessType | Existential | Max 1 |
| SSN:Process | SOS:hasSubProcess | SSN:Process | Universal | Nil |
| SSN:Sensing | SOS:hasControlStructure | SOS:ControlStructure | Universal | Min 1 |
| SSN:Sensor | SOS:recognizeContext | SOSS:Context | Universal | Min 1 |
| SSN:FeatureOfInterest | SOS:isContainedIn | SSN:Observation | Exis. & univ. | Min 1 |
| SSN:"Observation Value" | SOS:identifyContext | SOS:Context | Exis. & univ. | Min 1 |
| SOS:Context | SOS:hasTheme | SOS:Theme | Exis. & univ. | Min 1 |
| CXT:User ≡ SSN:Person | SOS:hasPreferenceProfile | SOS:PreferenceProfile | Exis. & univ. | Nil |
| SOS:Event | SOS:attende | CXT:User ≡ SSN:Person | Exis. & univ. | Min 1 |
| CXT:Location | SOS:hasFeatures | geonames:GeonamesFeatures | Universal | Nil |
| CXT:Activity | SOS:hasActivityLocation | CXT:Location | Exis. & univ. | Min 1 |

*4.1. Gold-Standard-Based Evaluation.* SmartOntoSensor is the first attempt to ontologically model smartphone sensory data and its context mapping. Therefore, no counterpart exists in the domain for comparison with SmartOntoSensor. However, a number of ontologies have been developed for sensors and sensor networks that include SSN [8], CSIRO [7], OntoSensor [5, 10], and CESN [12], which are mostly quoted in the literature. Therefore, SmartOntoSensor is compared with them to provide insights into its quality. Metrics and automated tools are defined for evaluating quality of ontology in some recent works such as OntoQA [33, 34]. OntoQA is feature-based ontology quality evaluation and analysis tool that has the capabilities of evaluating ontology at both schema and knowledge base levels. OntoQA uses a set of metrics to evaluate quality of ontology from different aspects including number of classes and properties, relationships richness, attributes richness, and inheritance richness. To evaluate the SmartOntoSensor quality, OntoQA is used. However, the evaluation and comparison are limited to schema level only because of the unavailability of knowledge bases of the competing ontologies. The overall comparison of SmartOntoSensor with the competing ontologies using OntoQA schema metrics is shown in Table 7.

Using the information provided in [33, 34], the interpretation of results using schema metrics indicates that SmartOntoSensor has improved ontology design with rich potential for knowledge representation as compared to the existing ontologies. SmartOntoSensor provides enhanced coverage of its broader modeling domain by having larger number of classes and relationships in comparison to existing ontologies. This also indicates that SmartOntoSensor is more complete by appropriately covering the problem domain by providing answers to almost any of the ontology domain-related questions. OntoSensor also has shown tremendous classes and relationships measurements; however, OntoSensor mainly describes the spectrum of sensors concepts (i.e., hierarchy of sensors classes and subclasses) and data. SmartOntoSensor, on the other hand, includes an extensive list of classes and relationships for describing broad aspects of smartphone systems, sensors, observation measurements, and context applications. The highest relationship richness value shows that SmartOntoSensor has diversity in types of relations in the ontology. Instead of relying only on inheritance relationships (usually conveying less information), SmartOntoSensor contains diverse set of relationships to convey almost complete information about the domain. The slight richness in relationships of SmartOntoSensor over CSIRO can be microscopically viewed large due to large number of SmartOntoSensor classes as compared to CSIRO. The increased number of relationships and relationship

Table 5: An excerpt of SmartOntoSensor datatype properties.

| Domain | Datatype property | Range | Quantifier | Cardinality |
|---|---|---|---|---|
| SOS:Manufacturer | SOS:manufacturerName | String | Exist. & univ. | Min 1 |
| SOS:SerialNumber | SOS:serialNumber | String | Existential | Exactly 1 |
| SOS:Size | SOS:weight | Float | Universal | Max 1 |
| SSN:“Sensor Output” | SOS:isValid | Boolean | Existential | Exactly 1 |
| SOS:Smartphone | SOS:hasIMEI | String | Existential | Exactly 1 |
| SOS:Smartphone | SOS:isConsumable | Boolean | Universal | Exactly 1 |
| SOS:MemoryResource | SOS:hasMemorySize | String | Existential | Exactly 1 |
| SOS:PowerSupplyResource | SOS:hasPowerCapacity | String | Existential | Exactly 1 |
| SOS:Version | SOS:versionNumber | String | Existential | Exactly 1 |
| SOS:Offset | SOS:accXAxis | Float | Existential | Exactly 1 |
| SOS:SpaceTuple | SOS:latitude | Float | Existential | Exactly 1 |
| SOS:Orientation | SOS:yaw | Float | Existential | Exactly 1 |
| SOS:PersoanlProfile | SOS:homePage | String | Universal | Nil |
| SOS:ContactProfile | SOS:phone | String | Universal | Max 1 |
| ontology:Location | SOS:officialName | String | Exist. & univ. | Min 1 |
| SOS:QualityValue | SOS:value | Float | Existential | Exactly 1 |
| SSN:“Unit of measure” | SOS:unit | String | Existential | Exactly 1 |
| SOS:DataContainer | SOS:path | String | Existential | Exactly 1 |
| Ontology:Humidity | SOS:hasHumidityIntensity | String | Existential | Exactly 1 |
| ontology:Lighting | SOS:hasLightIntensity | String | Existential | Exactly 1 |
| SOS:SmartphoneMetadata | SOS:hasDescription | String | Universal | Nil |
| SOS:DataContainer | SOS:hasName | String | Existential | Exactly 1 |
| SOS:ContactProfile | SOS:mobile | PositiveInteger | Universal | Nil |

richness also advocate high cohesiveness of SmartOntoSensor by strongly and intensively relating classes in the ontology. The lowest inheritance richness value proves SmartOntoSensor as a vertical ontology, which is covering the domain in a detailed manner. In other words, SmartOntoSensor is concise by not having irrelevant concepts or redundant representations of the semantics regarding the domain. The large number of classes and relationships in SmartOntoSensor makes its slight difference in inheritance richness much bigger in comparison with other available ontologies. The lowest tree balance value indicates that SmartOntoSensor can be more viewed as a tree compared to others. The highest class richness value of SmartOntoSensor indicates increased distribution of instances across classes by allowing knowledge base to utilize and represent most of the knowledge in the schema. In other words, most of the SmartOntoSensor classes would be populated with instances as compared to the other. Comparatively, SmartOntoSensor is ranked the highest due to its larger schema size in terms of number of classes and relationships. OntoQA cannot directly calculate coupling measure of ontology. However, by using the coupling definition [31], SmartOntoSensor also shows high coupling by referencing an increased number of classes from the imported ontologies.

*4.2. Multicriteria Approach Based Evaluation.* To evaluate semantics and understandability of terms used in ontology, researchers have not established any common consensus on widely accepted methods (i.e., tools and metrics) in computer science (i.e., it could be due to relatively being a new field) so far. However, to evaluate ontology terminologies along with its popularity, an approach comprising several decision criteria or attributes is defined by [35] where numerical scores can be assigned to each criterion. The overall score is calculated by the weighted sum of its per-criterion scores [29]. An objective formula for terminology evaluation and popularity measurement (shown in (1)) has been used from [35] that is based on the objective multicriteria matrices defined in [36].

$$\text{Objective} = I * w_i + C * w_c + O * w_o + P * w_p. \quad (1)$$

In (1), $I$ shows interoperability and is $I = N/T$, where $T$ is total number of terms in ontology and $N$ is the number of terms findable in WordNet. $C$ is clarity and is $C = (\sum 1/A_i)/N$, where $A_i$ shows the number of meanings of every interoperable term in WordNet, then clarity of each term is $1/A_i$, $O$ represents comprehensiveness, and is $O = T/M$, where $M$ is the number of terms in standard term set in the domain that the ontology belongs to. $P$ denotes popularity and is $P = E/H$, where $E$ denotes the number of access instances of the ontology and $H$ is the total number of access instances of all the ontologies in the same domain. The weights $w_i$, $w_c$, $w_o$, and $w_p$ represent the weight of interoperability, clarity, comprehensiveness, and popularity, respectively, with condition that these weights satisfy the equation $w_i + w_c + w_o + w_p = 1$. WordNet is a large lexical database of English words that is linking words by semantic relationships and has been used by several of the ontology

Table 6: Top-level terminological requirements fulfillment (using availability of concepts) comparison of SmartOntoSensor and other sensors and sensor networks ontologies.

| Terminological requirements category | Top-level concepts | Ontologies | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Avancha et al. [11] | OntoSensor [5, 10] | CESN [12] | Eid et al. [13] | CSIRO [7] | SSN [8] | ISTAR [14] | Kim et al. [15] | SmartOntoSensor |
| Base terms | System | | | | | | √ | √ | | √ |
| | Sensor | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| | Components/resources | √ | √ | | | | | | | √ |
| | Process | | √ | | | √ | √ | | | √ |
| | Context | | | | | | | | | √ |
| System terms | Platform | | | | | | | | | |
| | Hardware | | √ | | | √ | √ | √ | | √ |
| | Software | | | | | | | | | √ |
| | Deployment | | √ | √ | | | √ | √ | | √ |
| | System metadata | | | | | | | | | |
| | Identification | | | | | | | | | √ |
| | Manufacturer | | | | | | | | | √ |
| | Components/resources | | | | | | | | | |
| | Power supply | √ | √ | | | √ | | | | √ |
| | CPU | √ | | | | | | | | √ |
| | Memory | √ | | | | | | | | √ |
| | Networking | √ | √ | | | | | | | √ |
| Sensor terms | Sensor hierarchy | | | | | | | | | |
| | Physical sensor | | √ | √ | | √ | | √ | | √ |
| | Logical sensor | | | | | | | | | √ |
| | Type of operation | | | | | | | | | |
| | Active sensor | | √ | | | √ | | | | √ |
| | Passive sensor | | √ | | | √ | | | | √ |
| | Operating condition | √ | | | | √ | | √ | | √ |
| | History | | √ | | | | | | | |
| | Sensor metadata | | | | | | | | | |
| | Identification | | | | √ | √ | | √ | | √ |
| | Manufacturer | | | | √ | √ | | √ | √ | √ |
| | Configuration | | √ | | √ | | | | | √ |
| | Sensor process | | | | | | | | | |
| | Sensing & process type | √ | √ | | | √ | √ | | √ | √ |
| | Process parameters | √ | √ | | √ | √ | √ | | | √ |
| | Measurement properties | | | | | | | | | |
| | Frequency | | | | √ | | √ | | √ | √ |
| | Latency | | | | | √ | √ | | | √ |
| | Accuracy | √ | | | √ | √ | √ | | √ | √ |
| | Resolution | √ | | | | √ | √ | | | √ |
| | Measurement range | | | | √ | √ | √ | | | √ |
| | Precision | | | | | | √ | | | √ |
| | Power consumption | √ | | | | | | | | √ |
| | Sensitivity | | | | | | √ | | | √ |

Table 6: Continued.

| Terminological requirements category | Top-level concepts | Ontologies<br>Avancha et al. [11] | OntoSensor [5, 10] | CESN [12] | Eid et al. [13] | CSIRO [7] | SSN [8] | ISTAR [14] | Kim et al. [15] | SmartOntoSensor |
|---|---|---|---|---|---|---|---|---|---|---|
| Observation terms | Data/observation | √ | √ | √ | √ | √ | √ |  | √ | √ |
|  | Response model | √ | √ |  |  | √ |  |  |  | √ |
|  | Observation condition |  |  |  |  |  | √ |  |  | √ |
| Domain terms | Unit of measurement | √ | √ |  | √ | √ |  |  |  | √ |
|  | Feature/quality | √ | √ | √ | √ | √ | √ |  | √ | √ |
|  | Sampled medium | √ | √ |  |  |  | √ |  |  | √ |
| Context terms | Location | √ | √ | √ | √ | √ |  |  | √ | √ |
|  | Time |  | √ | √ |  |  |  |  |  | √ |
|  | Activity |  |  |  |  |  |  |  |  | √ |
|  | Event |  |  |  |  |  |  |  |  | √ |
|  | User |  |  |  |  |  |  |  |  | √ |
|  | Service |  |  |  |  |  |  |  |  | √ |
| Storage terms | File |  |  |  |  |  |  |  |  | √ |
|  | Folder |  |  |  |  |  |  |  |  | √ |

Table 7: Statistics of SmartOntoSensor and other sensors and sensor networks ontologies using OntoQA.

| Ontology | Classes | Relationships | Relationships richness | Inheritance richness | Tree balance | Class richness | Rank |
|---|---|---|---|---|---|---|---|
| SSN [8] | 47 | 52 | 59.09 | 2.4 | 1.76 | 66.18 | IV |
| OntoSensor [5, 10] | 286 | 219 | 46.39 | 2.63 | 1.66 | 59.37 | II |
| CESN [12] | 35 | 18 | 39.13 | 2.54 | 1.53 | 71.01 | V |
| CSIRO [7] | 70 | 70 | 65.42 | 2.84 | 1.36 | 39.65 | III |
| SOS | 259 | 382 | 65.52 | 2.23 | 1.19 | 84.67 | I |

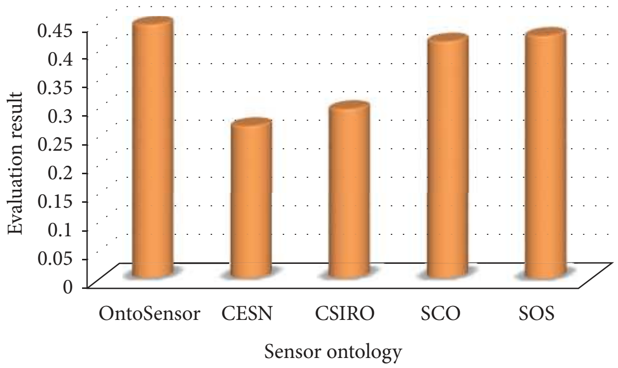


Figure 6: Objective analysis of SmartOntoSensor using multicriteria approach.

development researches for terminology definition, analysis, and filtering such as [34]. Using the guideline from [34], WordNet is used for the evaluation of terms interoperability, clarity, and comprehensiveness, and literature citation index is used for evaluation of popularity. During analysis, multiwords terms in SmartOntoSensor are broken into individual words and stemmed to receive more accurate metrics values. Results obtained by [35] using the same formula are extended by the inclusion of SmartOntoSensor results and shown in Figure 6.

Analyzing results indicates that SmartOntoSensor has acceptable levels of terms interoperability and comprehensiveness, medium clarity, and lowest popularity. Overall, SmartOntoSensor shows reasonable analysis results, signifying superiority over the others due to rich set of terms (i.e., classes and properties) but slightly less than OntoSensor due to lowest popularity value. Certainly, care has been taken while calculating the results, but the approach involves human efforts (due to unavailability of automated tools), where the chances of errors in the process cannot be ignored.

*4.3. Accuracy Checking.* To demonstrate the correctness and utility of SmartOntoSensor, accuracy checking is performed to determine that the asserted knowledge in the ontology agrees with the experts' knowledge in the domain. Ontology with correct definitions and descriptions of classes, properties, and individuals will result into high accuracy [31]. Recall and precision rates are the two primary Information Retrieval (IR) measures that are used for evaluating the accuracy of ontology [31]. Recall and precision rates are defined and shown in (2) and (3), respectively [13], and SmartOntoSensor is required to maximize both recall and precision rates for its acceptability.

$$\text{recall rate} = \left(\frac{\text{number of relevent items retrieved}}{\text{total number of relevant items}}\right), \quad (2)$$

$$\text{precision rate} = \left(\frac{\text{number of relevent items retrieved}}{\text{total number of items retrieved}}\right). \quad (3)$$

Accuracy of SmartOntoSensor is determined by computing the recall and precision rates for the functional requirements (represented in competency questions) by executing the SPARQL queries on the SmartOntoSensor knowledge base. To form knowledge base, SmartOntoSensor is instantiated with relevant information from USC Human Activity Dataset (USC-HAD) (http://www-scf.usc.edu/~mizhang/ http://www-scf.usc.edu/~mizhang/), manuals, reports, and documentations using Protégé 4.3. Based on these instances, SPARQL query language is used through Protégé SPARQL Query plug-in to query knowledge base for retrieving relevant results. The scenarios (built using competency questions) used for proof-of-concept assume utilizing low-level sensory data of heterogeneous sensors for mapping them to high-level queries. For example, a possible source of detecting and monitoring signatures of human fall is tracking vector forces exerted during fall and location changes. Therefore, mapping fall concept to a concept which could be determined by accelerometer, gyroscope, and GPS sensors through relationship SOS:isDetectedBy can enhance human fall detection. The scenarios used for proof-of-concept include (1) using microphone and GPS low-level sensory data to search locations having high noise pollutions and (2) using low-level sensory data to detect a user's context and automatically initiate a respective service (i.e., application). The actual running SPARQL queries for each of the scenarios are shown in Algorithms 1 and 2, respectively.

Using assistance of low-level sensory data, the testing queries resulted in acceptable precision and recall rates by retrieving relevant data (i.e., all of the locations having greater sound levels and all of the sensors having measurement capabilities of acceptable levels). Therefore, it has been observed that SmartOntoSensor is of potential effectiveness of integrating heterogeneous sensory data to answer high-level queries.

*4.4. Consistency Checking.* OWL-DL is used as the knowledge representation language for SmartOntoSensor content.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX sos: <http://www.semanticweb.org/shoonikhan/ontologies/2015/9/SmartOntoSensor#>
SELECT ?name ?latitude ?longitude
    WHERE {     ?mic sos:hasMicrophoneValue ?output.
                ?output sos:hasSoundOutput ?soundvalue.
                ?output sos:relatedOutput ?GPSoutput.
                ?GPSoutput sos:latitude ?latitude.
                ?GPSoutput sos:longitude ?longitude.
                ?GPSoutput sos:isCoordinatesOf ?location.
                ?location sos:officialName ?name.
                FILTER (?soundvalue > "65.0"^^xsd:float)
            }
```

Algorithm 1: SPARQL query for retrieving locations having noise intensity greater than 65 dB.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ssn: <http://purl.oclc.org/NET/ssnx/ssn#>
PREFIX cxt: <http://a.com/ontology#>
PREFIX sos: <http://www.semanticweb.org/shoonikhan/ontologies/2015/9/SmartOntoSensor#>
SELECT ?context ?service
    WHERE {       {
                  ?acc sos:hasAccelerometerValue ?offset.
                  ?offset sos:accXAxis ?xaxis.
                  ?offset sos:accYAxis ?yaxis.
                  ?offset sos:accZAxis ?zaxis.
                  FILTER (((?xaxis >= "-10.9"^^xsd:float) && (?xaxis <= "0.4"^^xsd:float)) &&
                   ((?yaxis >= "-0.5"^^xsd:float) && (?yaxis <= "0.6"^^xsd:float)) &&
                   ((?zaxis >= "-15.0"^^xsd:float) && (?zaxis <= "18.0"^^xsd:float))).
                  ?offset ssn:hasValue ?obsvalue.
                  ?obsvalue sos:hasObservationLocation ?obsloc.
                  ?obsvalue sos:identifyContext ?context.
                  ?context sos:hasActivityLocation ?cxtloc.
                  ?cxtloc sos:hasCoordinates ?activityloc.
                   }
                  FILTER (?obsloc = ?activityloc).
                  ?context sos:startService ?service.
            }
```

Algorithm 2: SPARQL query for detecting a context and service using low level sensory data.

A significant feature of the ontologies described using OWL-DL is that they can be processed using reasoners [13, 37]. Using the descriptions (conditions) of classes, a reasoner can determine whether a class can have an instance or not. Using the methodology proposed by [36] to validate consistency of ontology, SmartOntoSensor is passed through two major tests: (1) subsumption test to check whether a class is a subclass of another class or not and (2) logical consistency check to see whether a class can have any instance or not. Fact++ 1.6.2 and RacerPro 2.0 reasoners are used because of their strong reasoning capabilities and interoperability with Protégé [13]. Both of the reasoners are fed with manually created class hierarchy (called asserted ontology) to automatically compute an inferred class hierarchy (called inferred ontology) by using the descriptions of classes and relationships. The inferred ontology is compared with asserted ontology and found that both of the class hierarchies match and none of the classes is inconsistent. However, the classes are classified and repositioned by the reasoners in case of either having many super-classes or being subjected to some logical constraints. Therefore, both of the tests are significant and SmartOntoSensor is logically consistent and

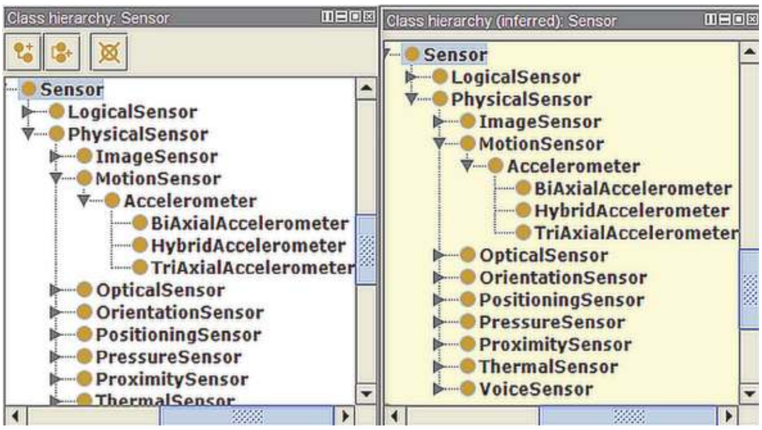


Figure 7: SmartOntoSensor asserted and inferred class hierarchies after using reasoner.

valid. Figure 7 depicts a snippet of the SmartOntoSensor asserted and inferred class hierarchies after reasoning.

*4.5. Application-Based Evaluation.* Ontologies can be plugged into applications, which would largely influence outputs of an application for a given task. Therefore, ontologies can be simply evaluated by analyzing results of using applications. In the coming sections, we have demonstrated a general architecture for context-aware applications that uses SmartOntoSensor and the development of a real-world smartphone context-aware application using the architecture.

*4.5.1. Application Architecture.* To demonstrate the feasibility and verifiability of SmartOntoSensor in real-world context-aware applications, we have developed a multilayer architecture shown in Figure 8. The architecture comprises four layers: sensors layer, information layer, semantic layer, and application layer, respectively, which are concisely described in the following headings.

*(1) Sensors Layer*. The sensors layer comprises smartphone sensors (i.e., physical and logical) that would provide not only raw sensory data from environment but also additional related information for effective context recognition such as BlutoothID of a nearby object to determine location and proximity of a user.

*(2) Information Layer*. The information layer extracts and receives raw sensory data and processes it into information. Information layer consists of two sublayers: collection engine and data aggregation and association. The collection engine acts as an interface between the sensors layer and data aggregation and association layer and contains several components. Sensor configuration defines sensors management activities (i.e., defining reading rate, etc.) and sensor acquisitor defines methods of reading data from sensors such as event-based or polling-based and from single sensor or multiple sensors in parallel. Sensors data processing extracts meaningful features from the sensors streams and uses capturing engine to store the sensory and other contextual information in a local temporary storage. The layer can also implement machine learning techniques to extract fine-grained contextual information out of sensory information. Data is temporarily stored locally to define a complete set of data about an event. The data aggregator and association collects data about a context from storage and establishes relationships with other cooccurring contexts. This layer clusters the data into context and subcontext groups and also extracts data from other sources (e.g., calendar entry for naming a context) to portray complete picture of a context. The complete set of data is converted into an exchangeable format (e.g., JSON) for communication to other layers.

*(3) Semantic Layer*. The semantic layer adds semantic glue to the architecture by mapping low-level context features into high-level semantic model (i.e., ontology). The semantic rule mapping and extension sublayer implements a web service interface to receive and extract information from well-formed message (i.e., JSON). The sublayer uses direct approach to define mapping and semantic rules to instantiate individuals in the SmartOntoSensor ontology and annotate them with their datatype and object properties and define relationships between individuals within the ontology, respectively. The direct approach is favorable because of its simplicity, whereas generic approach is complex requiring multiplicity of resources (i.e., vocabularies, classification algorithms, and domain-specific ontologies). However, the architecture supports generic approach by providing the required resources either at the information layer or at semantic layer. In case of direct approach, new rules are needed to be defined for new concepts. Rules are defined in event-condition-action format. For example, see Algorithm 3.

The semantic engine framework denotes a Semantic Web framework (e.g., AndroJena for Android-based smartphones) that provides the capabilities of hosting ontologies (e.g., SmartOntoSensor) and inferencing and reasoning mechanism to detect inconsistency in ontologies and deduces new knowledge on the basis of existing ones, features for updating ontologies with new individuals and annotations and extending ontologies by defining new classes and properties, triple store for storing data in RDF format,

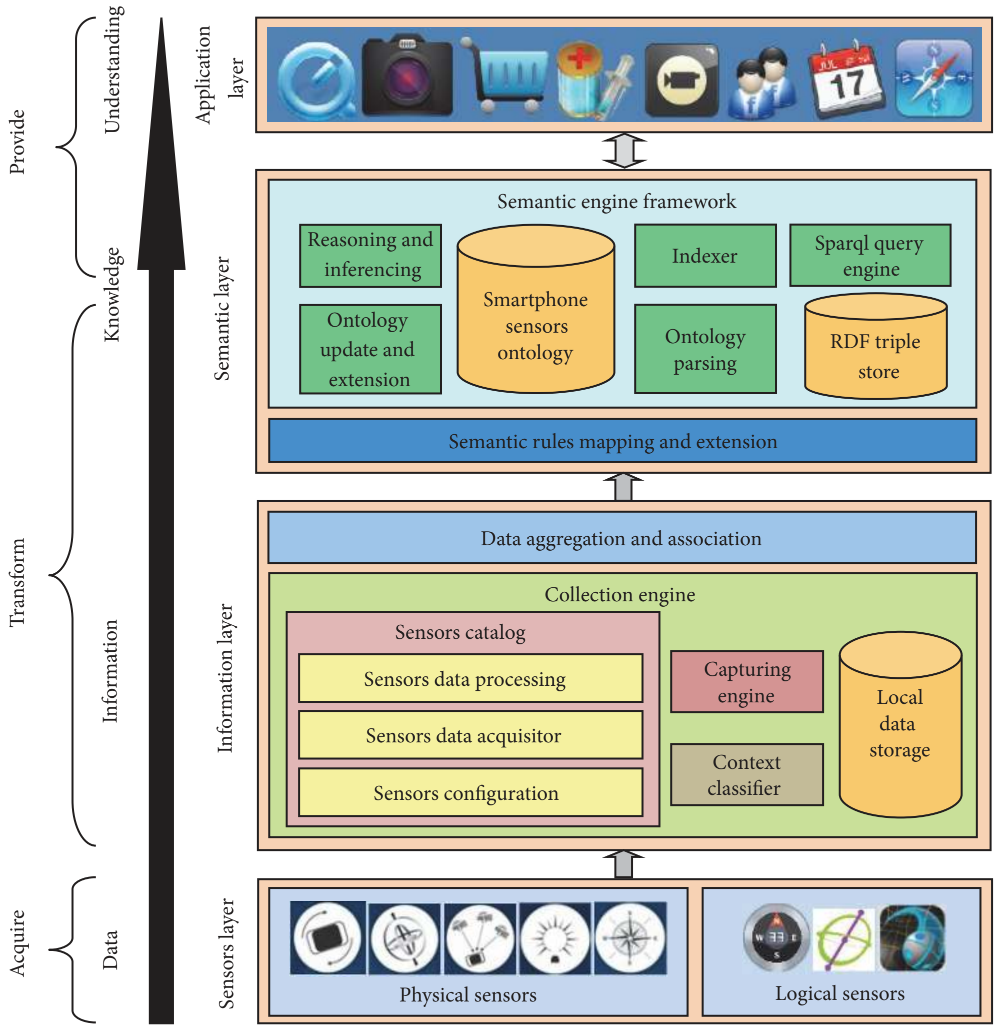


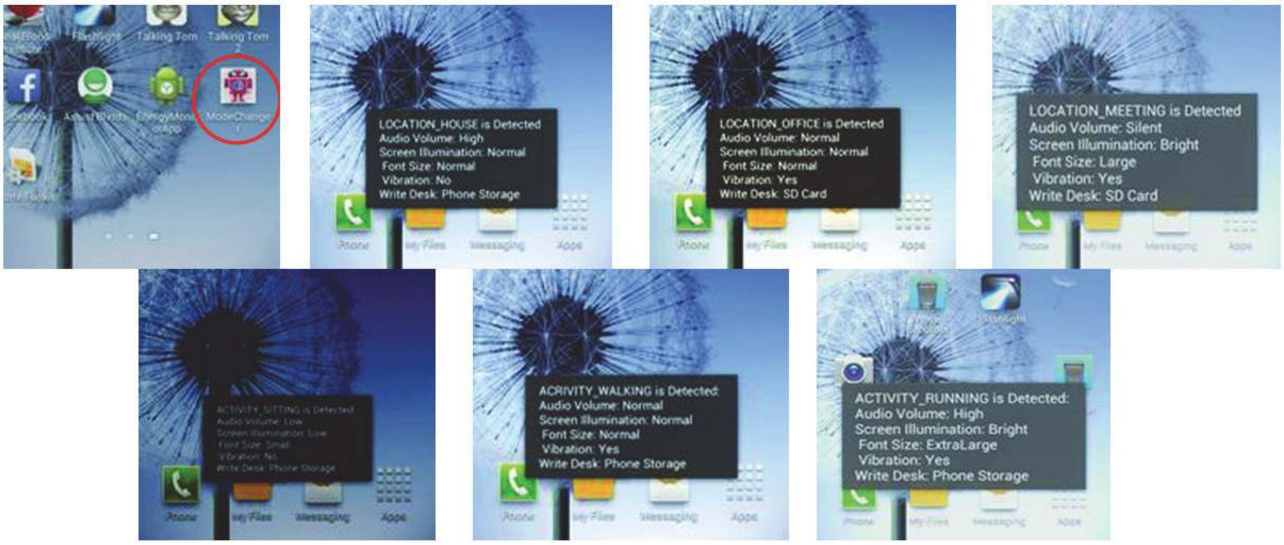
Figure 8: Architecture for smartphone context-aware applications using SmartOntoSensor.

```
IF EXIST event/sensors/GPS THEN
{
        CREATE INDIVIDUAL OF CLASS GPS IN
        SMARTONTOSENSOR WITH NAME event/sensors/GPS/@locationname
        IF EXIST event/sensors/GPS/@LatitudeValue THEN
              {
                     SET DATATYPE PROPERTY hasLatitude TO
                     event/sensors/GPS/@LatitudeValue
              }
        IF EXIST event/sensors/GPS/@LongnitudeValue THEN
              {
                     SET DATATYPE PROPERTY hasLongnitude TO
                     event/sensors/GPS/@LongnitudeValue
              }
}
```

Algorithm 3: Example of event-condition-action format.

Table 8: Hypothetical exemplary contexts and their corresponding modes values.

| Context | Audio volume | Screen brightness | Font size | Vibration | Write disk |
|---|---|---|---|---|---|
| Location:House | High | Normal | Normal | No | Phone storage |
| Location:Office | Normal | Normal | Normal | Yes | SD card |
| Location:Meeting | Silent | Bright | Normal | Yes | SD card |
| Activity:Sitting | Low | Low | Small | No | Phone storage |
| Activity:Walking | Normal | Normal | Normal | Yes | Phone storage |
| Activity:Running | High | Bright | Extra large | Yes | Phone storage |



LOCATION_HOUSE is Detected
Audio Volume: High
Screen Illumination: Normal
Font Size: Normal
Vibration: No
Write Desk: Phone Storage

LOCATION_OFFICE is Detected
Audio Volume: Normal
Screen Illumination: Normal
Font Size: Normal
Vibration: Yes
Write Desk: SD Card

LOCATION_MEETING is Detected
Audio Volume: Silent
Screen Illumination: Bright
Font Size: Large
Vibration: Yes
Write Desk: SD Card

ACTIVITY_SITTING is Detected
Audio Volume: Low
Screen Illumination: Low
Font Size: Small
Vibration: No
Write Desk: Phone Storage

ACRIVITY_WALKING is Detected:
Audio Volume: Normal
Screen Illumination: Normal
Font Size: Normal
Vibration: Yes
Write Desk: Phone Storage

ACTIVITY_RUNNING is Detected:
Audio Volume: High
Screen Illumination: Bright
Font Size: ExtraLarge
Vibration: Yes
Write Desk: Phone Storage

Figure 9: Changing modes by ModeChanger using contextual information from SmartOntoSensor.

and SPARQL query engine for dealing with users' initiated SPARQL queries.

*(4) Application Layer*. The application layer represents the context-aware applications, which can exploit the full potential of the architecture in general and SmartOntoSensor in specific to automatically recognize context and initiate services accordingly. The architecture provides a composite framework, which can be used by a wide variety of developers for developing any type of context-aware applications. As with the passage of time lifestyles changes and new types of data sources (i.e., sensors) can emerge, new mapping and semantic rules are needed to be defined accordingly. Mapping and semantic rules applications can be provided as either a separate application or an integral part of a context-aware application.

*4.5.2. ModeChanger Application*. Using the architecture shown in Figure 8, we have developed a prototype application called ModeChanger. A smartphone mode defines the number and nature of features, resources, and services available to consume at a time instant. Modern smartphones' operating systems support a number of operation modes that can be explicitly adjusted by users to define a smartphone's behavior according to a context. Currently, the available modes are airplane/flight mode, normal mode, audio mode, and brightness mode. The context-dependent mode changing can benefit from manipulating and accessing context information. The application can ease human-phone interaction and adjusts smartphone modes using contextual information. The prototype application is aimed for Android-based smartphones running with Ice Cream Sandwich 4.0.3 or higher and is developed in Java programming language using Android SDK tools revision 22.6.3 with Eclipse IDE and SensorSimulator-2.0-RC1 running on a desktop machine. The target code is deployed and tested on Samsung Galaxy SIII running with Android Jelly Bean 4.1.1 operating system. The application runs inconspicuously in the background as a service and utilizes SmartOntoSensor to deduce contextual information using low-level sensory data and automatically adjusts modes by invoking low-level services. The fuzzy logic controller is used to direct the overall adjustment process according to contexts. To adjust a smartphone's behavior, the application adjusts audio volume, screen brightness, font size, vibration, and default write disk according to the identified context. Table 8 lists a few hypothetical exemplary contexts and their corresponding modes values. The application is tested closely and found that the application can successfully differentiate between different user contexts in real-time by mapping low-level sensory data into high-level contexts and trigger services accordingly. Figure 9 presents screen shots of ModeChanger changing modes according to changing contexts.

*4.6. Experimental Method Base Evaluation.* Using the guidelines of [38], an empirical study is designed to evaluate the effectiveness of SmartOntoSensor. The empirical study is composed of experiments for evaluating SmartOntoSensor from the aspects: (1) requirements coverage, (2) goals and design characteristics, (3) explicitness and usability, (4) reusability, (5) stability, (6) modeling mistakes, and (7) ModeChanger application. The data for the empirical study is collected from the participants using a questionnaire. The total number of participants in the study is 17 having substantial experience and knowledge about ontology development and evaluation, knowledge of OWL, and problem domain. The participants are voluntarily selected from the master (M.S.) and Ph.D. students specializing in the areas of Web Semantics and Wireless Sensor Networks in the Department of Computer Science, University of Peshawar. However, to do justice with the study, one-day workshop on ontology development in general and SmartOntoSensor development in specific was arranged for the participants. Furthermore, the participants were provided the SmartOntoSensor source code and ModeChanger application was installed on their smartphones for observations, experiments, and practical usage for five days. At the end of the time, each participant (individually) filled out a questionnaire composed of 35 questions that are comprehensive enough to analyze SmartOntoSensor from the abovementioned aspects. Questions in the questionnaire are propositions whose answers are selected from a 5-level Likert-Scale, ranging from "strongly disagree" to "strongly agree." Table 9 represents statistical information of the participants' responses to the questions in percentage. About 71.4% of the participants (17.8% strongly agree and 53.6% agree) responses agreed in effectiveness of SmartOntoSensor and showed their confidence, 13.5% of the participants (3.9% strongly disagree and 9.6% disagree) responses did not, and 15.1% of the participants' responses remain neutral.

The Likert-Scale data is ordinal, where the order of the values is significant and important but the exact difference between the values is not really known. To analyze the ordered scale 5-level Likert-Scale responses data using Chi-Square descriptive statistics, the five response categories (i.e., strongly disagree, disagree, neutral, agree, and strongly agree) are broken down into two nominal categories (i.e., disagree and agree) by combining the lower level three categories and upper level two categories, respectively. Chi-Square is important statistic for analysis of categorical data. Table 10 represents the division of five-level Likert-Scale response categories into two nominal categories, percentage values in the nominal categories, and in the total. Chi-Square test is executed on the nominal categories using SPSSS 16.0 to show the effectiveness of SmartOntoSensor. The null and alternative hypotheses are, respectively, as follows:

($H_0$) SmartOntoSensor is not an effective ontology for smartphone-based context-aware computing.

($H_1$) SmartOntoSensor is an effective ontology for smartphone-based context-aware computing.

Results of the Chi-Square test are shown in Table 11. The top row in the table shows Pearson Chi-Square statistics $\chi^2 = 5.950E2$ and $p < 0.001$. The null hypothesis ($H_0$) is rejected, since $p < 0.05$ (i.e., in fact $p < 0.001$). Therefore, the alternative hypothesis ($H_1$) stands true, which signifies that SmartOntoSensor is an effective ontology for smartphone-based context-aware computing.

## 5. Applications of SmartOntoSensor

SmartOntoSensor is an attempt to provide semantic model by associating metadata with smartphone and sensory data to be used in potential context-aware smartphone applications. It has a broad spectrum of applications and can find place anywhere where smartphone sensors are employed. However, a few of the broad application areas of SmartOntoSensor could be as follows.

*5.1. Linked Open Data.* The Linked Open Data (LOD) uses Semantic Web technologies to provide an infrastructure for publishing structured data on the web regarding any domain in such a way that is formal and explicit and can be linked to or linked from external datasets [39] to increase its usefulness. Information captured in SmartOntoSensor can be linked with data sources on LOD including GeoData and DBpedia, and Internet of Thing (IoT) objects, which are discoverable and accessible through LOD cloud. The SmartOntoSensor concepts can be automatically annotated with information from other spatially and thematically related sensors to provide a scalable and semantically rich data model. This way information can be integrated from different communities and sources for a number of reasons including drawing conclusions, creating business intelligence, and automated decision making. The large-scale data generated by SmartOntoSensor and its linkage with LOD will result into Linked Big Sensors data providing a novel platform for publishing and consuming smartphone sensors data, which can be queried, retrieved, analyzed, reasoned, and inferred for solving real-world problems [39].

*5.2. Lifelogging.* Lifelogging is a type of pervasive computing that uses digital sensors to capture and archive a unified digital record of peoples' lifetime experiences in multimedia format for augmenting human memory. Researchers have shown smartphones as an ideal platform for potential lifelogging systems due to its technological advancements, sensing capabilities, and being a constant companion of users [40]. Lifelogging systems can use SmartOntoSensor for semantic annotation of captured lifelog information together with sensory information and contexts derived from low-level sensory data. SmartOntoSensor can allow for semantic-based reasoning on the captured lifelog data and their metadata for deducing new semantics and relationships among lifelog objects. Furthermore, the ontology can enhance retrieval of lifelog information for augmenting memory in real-time by allowing users to concisely express their queries and obtain precise answers using semantics of the contained data and queries. Lifelog information can be further enriched by exploiting the potential of SmartOntoSensor for linking and extracting data from data sources in the LOD cloud.

Table 9: Participants responses to the questions in questionnaire.

| Questions ∗ sample | Likert-Scale options | Frequency | Percent | Valid percent | Cumulative percent |
|---|---|---|---|---|---|
| 35 ∗ 17 | Strongly disagree | 23 | 3.9 | 3.9 | 3.9 |
| | Disagree | 57 | 9.6 | 9.6 | 13.4 |
| | Neutral | 90 | 15.1 | 15.1 | 28.6 |
| | Agree | 319 | 53.6 | 53.6 | 82.2 |
| | Strongly agree | 106 | 17.8 | 17.8 | 100.0 |
| | Total | 595 | 100.0 | 100.0 | |

Table 10: Division of 5-level Likert-Scale response categories into nominal categories and their percentage values within nominal categories.

| Nominal categories | Category wise distribution | Five levels of Likert-Scale response categories | | | | | Total |
|---|---|---|---|---|---|---|---|
| | | Strongly disagree | Disagree | Neutral | Agree | Strongly agree | |
| Disagree | Count | 23 | 57 | 90 | 0 | 0 | 170 |
| | % value within disagree | 13.5% | 33.5% | 52.9% | .0% | .0% | 100.0% |
| Agree | Count | 0 | 0 | 0 | 319 | 106 | 425 |
| | % value within agree | .0% | .0% | .0% | 75.1% | 24.9% | 100.0% |
| Total | Count | 23 | 57 | 90 | 319 | 106 | 595 |
| | % value within total | 3.9% | 9.6% | 15.1% | 53.6% | 17.8% | 100.0% |

Table 11: Results of the Chi-Square test using SPSS 16.0.

| Descriptive statistics | Value | df | $p$ value |
|---|---|---|---|
| Pearson Chi-Square | $5.950E2^{a}$ | 4 | .000 |
| Likelihood ratio | 711.941 | 4 | .000 |
| Linear-by-linear association | 425.035 | 1 | .000 |
| $N$ of valid cases | 595 | | |

[a] 0 cells (.0%) have expected count less than 5. The minimum expected count is 6.57.

*5.3. Smart Environment.* In a smart environment, smartphones can serve as smart objects because of their small ubiquitous nature, having sensor capabilities to detect environment, and communication capabilities to communicate with other objects through local networks or the Internet under the Internet of Things (IoT) domain in order to create a machine to machine ecosystem [41]. The increasing use and need for personal smart objects such as indoor or outdoor sensors and actuators have created the need to develop applications which a user may use to log and interact with these devices under the prism of the Internet or IoT. The SmartOntoSensor ontology can be a key for the automatic smart environment applications. Smartphone applications can leverage SmartOntoSensor for recording and mapping sensory data captured from smartphone sensors and sensors in an environment and transferring the captured and inferred data to a web database that could be shared with others (i.e., friends in a social network, etc.). Smartphone smart applications can use SmartOntoSensor for inferring users' contexts and automatically directing/controlling smart objects in environment accordingly. Similarly, SmartOntoSensor can be important resource for Google's Android at Home [42] that uses IoT for allowing individuals to log into their accounts and control their smart objects at home.

*5.4. Activity Recognition.* Smartphones offer a unique opportunity to sense and recognize human activities from location, proximity, and communication data [43]. Making a smartphone aware of users' activities fits into the larger framework of context awareness. Researchers have demonstrated successful applications of statistical machine learning models for smartphone-based activity recognition. However, activity recognition systems, despite producing significant accuracy in recognizing activities, suffer from a number of problems including application of a small number of sensors, requiring an extensive amount of training data, and addressing a small set of coarse-grained activities. Similarly, some activities are hard to recognize due to either using type and number of sensors, duration of an activity, or statistical classification algorithms used [44]. Recognizing a large set of activities with minimal processing requirements is essential for satisfying the diverse applications of smartphone-based activity recognition systems. SmartOntoSensor, by capturing and fusing data obtained from on-board sensors and external sensors as well as sources, would enable us to infer a fine-grained classification and recognition of not only a large set of activities but hard-to-recognize activities as well with minimal processing requirements.

*5.5. Augmented Reality (AR).* AR is the combination/registration and alignment of virtual and physical objects as well as their interaction in three dimensions in real-time [45]. Smartphones can serve as a charming blend for AR systems due to their advanced computational capabilities, user interface, and sensors and provides feasibility of integrating all of the solutions in one place such as MARA and MIT sixth-sense project [45]. SmartOntoSensor can serve as a bottom layer for helping upper layer AR systems. Data contained in SOS can be used by AR system for numerous purposes including registration/authentication, localization, viewing, indoor/outdoor navigation guidance, context recognition, and annotating objects with sensory and contextual information.

## 6. Conclusion

The potential power of smartphone sensing has been realized due to widespread adoption of the sensor-enabled smartphone technologies by people across several demographics and cultures. Smartphone technological advancements and integration of high-valued sensors have pushed development of smart sensing applications by the research community, academia, and industry for solving real-world problems. However, the integration and utilization of huge amount of heterogeneous data generated by heterogeneous smartphone sensors need semantic representation as a prerequisite for their unambiguous and precise interpretation and advanced analytical processing. As a solution, ontology can be deemed as a lethal weapon by containing detailed definitions of concepts regarding smartphone and sensors as well as their properties to solve the integration, processing, interpretation, and interoperability issues. Ontology would assist smartphone context-aware applications in querying a sensor or group of sensors for extracting low-level data for performing high-level tasks. Although a number of sensors and sensor networks ontologies are presented by the researchers, none of them is complete enough to satisfy the requirements and usage of smartphones context-aware applications. However, some of these ontologies (e.g., SSN) can be reused to develop a comprehensive smartphone sensors ontology.

In this paper, the development of ontology-based smartphones and sensors repository referred to as SmartOntoSensor is presented. SmartOntoSensor includes definitions of concepts and properties that are partially extended from more general and comprehensive SSN ontology, partially influenced from SensorML, and partially identified from the relevant literature. SmartOntoSensor provides descriptions about smartphone systems and deployments, sensors and their classification, sensors measurement capabilities and properties, observations as properties of features of interests, inputs and outputs to sensing processes, context identification using sensors outputs, and invoking services according to contexts. To the best of our knowledge this is the first such effort in the area of smartphone sensors ontologies to unify smartphone, sensors, and contextual information with general world knowledge about entities and relations. SmartOntoSensor is developed and evaluated using state-of-the-art technologies and standards to demonstrate its utility and value. The test results indicate that SmartOntoSensor has an improved ontological design for semantic modeling of the domain as compared to the other ontologies and is more complete by providing rich potential for representing information about the domain to answer ontology-related questions. Using quality and efficiency of SmartOntoSensor, a number of its potential application areas were also outlined.

However, SmartOntoSensor does not claim feasibility of orthogonal and universally acceptable smartphone sensors ontology but is an attempt to build a pragmatic smartphone sensors repository with supporting rationale and using currently available tools for enabling the deployment of the ontology in a variety of application domains. Despite promising lab test results and proofs-of-concepts, SmartOntoSensor needs improvements in several aspects to claim its market place such as (1) inclusion of more detailed and relevant concepts and properties to increase its coverage and expressiveness, (2) heavily instantiating of the concepts with real-world data to thoroughly test its quality and correctness, (3) thorough investigation by the domain experts to indicate any discrepancy, redundancy, and ambiguity, and (4) knowledge-based comparison with sensors and sensor networks ontologies in addition to schema-based comparison.

The future work includes the application of SmartOntoSensor in more complex real-world solutions to check its efficiency and performance and indicate any potential extensions and improvements. In addition, most of the organizations have captured sensors collected information in their repositories in either structured or unstructured (e.g., GeoLife GPS Trajectories) formats. Therefore, there is a strong need for systems that can automatically analyze various structured and unstructured data sources and extract relevant concepts and entities to extend and populate SmartOntoSensor. This automatic information extraction process would help in further evaluating SmartOntoSensor and designing more effective context-aware applications.

## Competing Interests

The authors declare that they have no competing interests.

## Acknowledgments

This research work has been undertaken by the first author as a partial fulfillment of Ph.D. degree with support of the Higher Education Commission (HEC) of Pakistan.

## References

[1] S. Ali, S. Khusro, A. Rauf, and S. Mahfooz, "Sensors and mobile phones: evolution and state-of-the-art," *Pakistan Journal of Science*, vol. 66, no. 4, pp. 385–399, 2014.

[2] Y. Wang, J. Lin, M. Annavaram et al., "A framework of energy efficient mobile sensing for automatic user state recognition," in *Proceedings of the 7th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, pp. 179–192, Kraków, Poland, June 2009.

[3] J. Subercaze, P. Maret, N. M. Dang, and K. Sasaki, "Context-aware applications using personal sensors," in *Proceedings of the 2nd International Conference on Body Area Networks (ICST '07)*, pp. 1–5, Florence, Italy, 2007.

[4] P. Korpipää and J. Mäntyjärvi, "An ontology for mobile device sensor-based context awareness," in *Proceedings of the 4th International and Interdisciplinary Conference (CONTEXT '03)*, pp. 451–458, Stanford, Calif, USA, June 2003.

[5] D. J. Russomanno, C. R. Kothari, and O. A. Thomas, "Building a sensor ontology: a practical approach leveraging ISO and OGC models," in *Proceedings of the International Conference on Artificial Intelligence (ICAI '05)*, pp. 637–643, Las Vegas, Nev, USA, June 2005.

[6] C. A. Henson, J. K. Pschorr, A. P. Sheth, and K. Thirunarayan, "SemSOS: semantic sensor observation service," in *Proceedings of the International Symposium on Collaborative Technologies and Systems (CTS '09)*, Baltimore, md, USA, May 2009.

[7] H. Neuhaus and M. Compton, "The semantic sensor network ontology: a generic language to describe sensor assets," in *Proceedings of the AGILE 2009 Pre-Conference Workshop Challenges in Geospatial Data Harmonisation*, Hannover, Germany, 2009.

[8] M. Compton, P. Barnaghi, L. Bermudez et al., "The SSN ontology of the W3C semantic sensor network incubator group," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 17, pp. 25–32, 2012.

[9] N. F. Noy and D. L. McGuinness, "Ontology Development 101: A Guide to Creating Your First Ontology," 2001, http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness.pdf.

[10] D. J. Russomanno, C. Kothari, and O. Thomas, "Sensor ontologies: from shallow to deep models," in *Proceedings of the 37th Southeastern Symposium on System Theory (SST '05)*, pp. 107–112, Tuskegee, Ala, USA, March 2005.

[11] S. Avancha, C. Patel, and A. Joshi, "Ontology-driven adaptive sensor networks," in *Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous '04)*, pp. 194–202, Boston, Mass, USA, August 2004.

[12] M. Calder, R. A. Morris, and F. Peri, "Machine reasoning about anomalous sensor data," *Ecological Informatics*, vol. 5, no. 1, pp. 9–18, 2010.

[13] M. Eid, R. Liscano, and A. El Saddik, "A universal ontology for sensor networks data," in *Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA '07)*, pp. 59–62, IEEE, Ostuni, Italy, June 2007.

[14] M. Gomez, A. Preece, M. P. Johnson et al., "An ontology-centric approach to sensor-mission assignment," in *Knowledge Engineering: Practice and Patterns*, A. Gangemi and J. Euzenat, Eds., vol. 5268 of *Lecture Notes in Computer Science*, pp. 347–363, Springer, Berlin, Germany, 2008.

[15] J.-H. Kim, H. Kwon, D.-H. Kim, H.-Y. Kwak, and S.-J. Lee, "Building a service-oriented ontology for wireless sensor networks," in *Proceedings of the 7th IEEE/ACIS International Conference on Computer and Information Science (IEEE/ACIS ICIS '08)*, pp. 649–654, Portland, Ore, USA, May 2008.

[16] M. Botts and A. Robin, "OpenGIS® Sensor Model Language (SensorML) Implementation Specification," 2007 http://portal.opengeospatial.org/files/?artifact_id=21273.

[17] A. Pease, I. Niles, and J. Li, "The suggested upper merged ontology: a large ontology for the semantic web and its applications," in *Proceedings of the AAAI-2002 Workshop on Ontologies and the Semantic Web*, Edmonton, Canada, 2002.

[18] M. Compton, C. Henson, L. Lefort, H. Neuhaus, and A. Sheth, "A survey of the semantic specification of sensors," in *Proceedings of the 8th International Semantic Web Conference (ISWC '09), 2nd International Workshop on Semantic Sensor Networks*, Washington, DC, USA, 2009.

[19] A. Underbrink, K. Witt, J. Stanley, and D. Mandl, "Autonomous mission operations for sensor webs," in *Proceedings of the American Geophysical Union, Fall Meeting 2008*, San Francisco, Calif, USA, December 2008.

[20] L. Bermudez, E. Delory, T. O'Reilly, and J. Del Rio Fernandez, "Ocean observing systems demystified," in *Proceedings of the MTS/IEEE Biloxi—Marine Technology for Our Future: Global and Local Challenges (OCEANS '09)*, pp. 1–7, Biloxi, Miss, USA, October 2009.

[21] C. Schlenoff, T. Hong, C. Liu, R. Eastman, and S. Foufou, "A literature review of sensor ontologies for manufacturing applications," in *Proceedings of the 11th IEEE International Symposium on Robotic and Sensors Environments (ROSE '13)*, pp. 96–101, October 2013.

[22] Y. Shi, G. Li, X. Zhou, and X. Zhang, "Sensor ontology building in semantic sensor web," in *Internet of Things*, Y. Wang and X. Zhang, Eds., vol. 312, pp. 277–284, Springer, Berlin, Germany, 2012.

[23] M. C. Suárez-Figueroa, A. Gómez-Pérez, and M. Fernández-López, "The NeOn methodology for ontology engineering," in *Ontology Engineering in a Networked World*, M. C. Suárez-Figueroa, A. Gómez-Pérez, E. Motta, and A. Gangemi, Eds., pp. 9–34, Springer, Berlin, Germany, 2012.

[24] S. Ali, S. Khusro, and H. Chang, "POEM: practical ontology engineering model for semantic web ontologies," *Cogent Engineering*, vol. 3, no. 1, pp. 1–39, 2016.

[25] V. Presutti and A. Gangemi, "Content ontology design patterns as practical building blocks for web ontologies," in *Proceedings of the 27th International Conference on Conceptual Modeling*, pp. 128–141, Berlin, Germany, 2008.

[26] X. Wang, X. Zhang, and M. Li, "A survey on semantic sensor web: sensor ontology, mapping and query," *International Journal of u-and e-Service, Science and Technology*, vol. 8, no. 10, pp. 325–342, 2015.

[27] E. Simperl, "Reusing ontologies on the Semantic Web: a feasibility study," *Data & Knowledge Engineering*, vol. 68, no. 10, pp. 905–925, 2009.

[28] D. Preuveneers, J. V. den Bergh, D. Wagelaar et al., "Towards an extensible context ontology for ambient intelligence," in *Proceedings of the 2nd European Symposium on Ambient Intelligence*, pp. 148–159, Eindhoven, Netherlands, 2004.

[29] J. Brank, M. Grobelnik, and D. Mladenic, "A survey of ontology evaluation techniques," in *Proceedings of the 8th International Multi-Conference Information Society*, pp. 166–169, 2005.

[30] T. J. Lampoltshammer and T. Heistracher, "Ontology evaluation with Protégé using OWLET," *Infocommunications Journal*, vol. 6, pp. 12–17, 2014.

[31] H. Hlomani and D. Stacey, "Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: a survey," 2014, http://www.semantic-web-journal.net/system/files/swj657.pdf.

[32] A. Gomez-perez, *Ontology Evaluation*, Springer, Berlin, Germany, 1st edition, 2004.

[33] S. Tartir and I. B. Arpinar, "Ontology evaluation and ranking using OntoQA," in *Proceedings of the 1st IEEE International Conference on Semantic Computing (ICSC '07)*, pp. 185–192, Irvine, Calif, USA, September 2007.

[34] S. Tartir, I. B. Arpinar, and A. Sheth, "Ontological evaluation and validation," in *Theory and Applications of Ontology: Computer Applications*, R. Poli, M. Healy, and A. Kameas, Eds., pp. 115–130, Springer, Amsterdam, The Netherlands, 2010.

[35] Y. Shi, G. Li, X. Zhou, and X. Zhang, "Sensor ontology building in semantic sensor web," in *Internet of Things*, Y. Wang and X. Zhang, Eds., vol. 312, pp. 277–284, Springer, 2012.

[36] A. Burton-Jones, V. C. Storey, V. Sugumaran, and P. Ahluwalia, "A semiotic metrics suite for assessing the quality of ontologies," *Data and Knowledge Engineering*, vol. 55, no. 1, pp. 84–102, 2005.

[37] M. Horridge, H. Knublauch, A. Rector, R. Stevens, and C. Wroe, "A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools," 2004, http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf.

[38] E. Blomqvist, V. Presutti, E. Data, and A. Gangemi, "Experimenting with eXtreme design," in *Proceedings of the Proceedings of the 17th International Conference on Knowledge Engineering and Management by the Masses EKAW*, vol. 10, pp. 120–134, Lisbon, Portugal, 2010.

[39] S. Khusro, S. Ali, A. Rauf et al., "Unleashing sensor data on linked open data—the story so far," *Life Science Journal*, vol. 10, no. 4, pp. 1766–1786, 2013.

[40] C. Gurrin, Z. Qiu, M. Hughes et al., "The smartphone as a platform for wearable cameras in health research," *American Journal of Preventive Medicine*, vol. 44, no. 3, pp. 308–313, 2013.

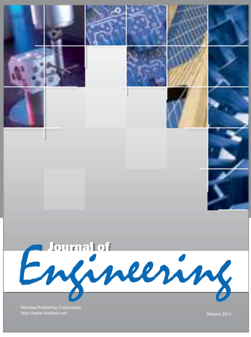
[41] N. E. Petroulakis, I. G. Askoxylakis, and T. Tryfonas, "Lifelogging in smart environments: challenges and security threats," in *Proceedings of the IEEE International Conference on Communications (ICC '12)*, pp. 5680–5684, Ottawa, Canada, June 2012.

[42] P. Wetterwald, "Android@Home," in *Proceedings of the Google I/O Developer Conference*, San Francisco, Calif, USA, 2011.

[43] D. Choujaa and N. Dulay, "Activity recognition using mobile phones: achievements, challenges and recommendations," in *Proceedings of the Workshop on How to Do Good Research in Activity Recognition: Experimental Methodology, Performance Evaluation and Reproducibility in Conjunction with UBICOMP*, 2010.

[44] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," in *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence*, vol. 3, pp. 1541–1546, Pittsburgh, Pennsylvania, July 2005.

[45] A. Khan, S. Khusro, A. Rauf, and S. Mahfooz, "Rebirth of augmented reality—enhancing reality via smartphones," *Bahria University Journal of Information & Communication Technologies*, vol. 8, no. 1, pp. 110–121, 2015.

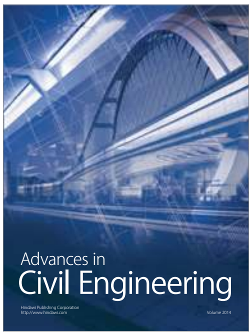

Journal of
Engineering

The Scientific World Journal

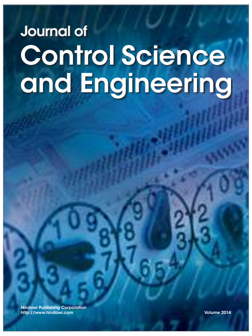
International Journal of
Rotating Machinery

Journal of
Sensors

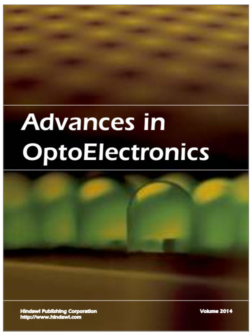
International Journal of
Distributed Sensor Networks


Advances in
Civil Engineering

Journal of
Robotics

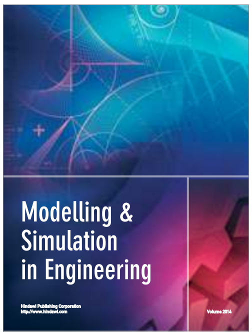

Journal of
Control Science and Engineering

Journal of
Electrical and Computer Engineering

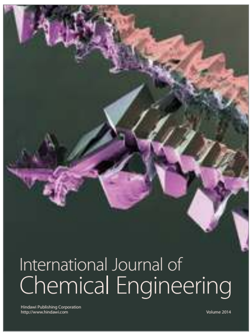

Advances in
OptoElectronics

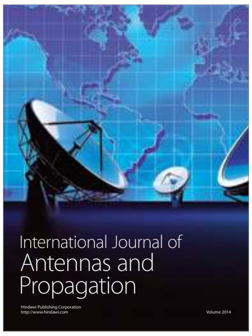
VLSI Design

International Journal of
Navigation and Observation

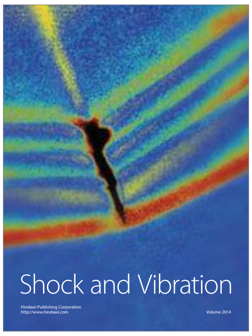

Modelling &
Simulation
in Engineering

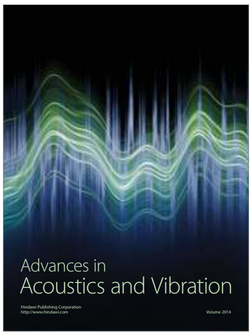
International Journal of
Aerospace Engineering


International Journal of
Chemical Engineering


International Journal of
Antennas and Propagation

Active and Passive
Electronic Components


Shock and Vibration


Advances in
Acoustics and Vibration