

Context-aware Sensor Search, Selection and Ranking Model for Internet of Things Middleware

Charith Perera^{*†}, Arkady Zaslavsky[†], Peter Christen^{*}, Michael Compton[†] and Dimitrios Georgakopoulos[†]

^{*}Research School of Computer Science, The Australian National University, Canberra, ACT 0200, Australia

[†]CSIRO ICT Center, Canberra, ACT 2601, Australia

Abstract—As we are moving towards the Internet of Things (IoT), the number of sensors deployed around the world is growing at a rapid pace. Market research has shown a significant growth of sensor deployments over the past decade and has predicted a substantial acceleration of the growth rate in the future. It is also evident that the increasing number of IoT middleware solutions are developed in both research and commercial environments. However, sensor search and selection remain a critical requirement and a challenge. In this paper, we present CASSARAM, a context-aware sensor search, selection, and ranking model for Internet of Things to address the research challenges of selecting sensors when large numbers of sensors with overlapping and sometimes redundant functionality are available. CASSARAM proposes the search and selection of sensors based on user priorities. CASSARAM considers a broad range of characteristics of sensors for search such as reliability, accuracy, battery life just to name a few. Our approach utilises both semantic querying and quantitative reasoning techniques. User priority based weighted Euclidean distance comparison in multidimensional space technique is used to index and rank sensors. Our objectives are to highlight the importance of sensor search in IoT paradigm, identify important characteristics of both sensors and data acquisition processes which help to select sensors, understand how semantic and statistical reasoning can be combined together to address this problem in an efficient manner. We developed a tool called CASSARA to evaluate the proposed model in terms of resource consumption and response time.

Index Terms—Internet of Things, context awareness, IoT middleware, sensors, sensor discovery, search and selection, sensor indexing and ranking, semantic and probabilistic reasoning, querying, multidimensional data fusion.

I. INTRODUCTION

The numbers of sensors deployed around the world are increasing at a rapid pace. These sensors continuously generate enormous amounts of data. Collecting and storing data from all the available sensors may not create additional value or solve the problem of efficient sensor data processing. Further, it may not be feasible due to large scale, resource limitations, and cost factors. When significant amounts of sensors are available to choose from, it becomes a challenge and a time consuming task to select the *appropriate* sensors. We describe the term *appropriate* in section III.

Sensing as a service (SensaaS) model is expected to build on top of the IoT infrastructure and services. SensaaS model envisions that sensors and/or sensor data streams would be available to use over the Internet following some utility arrangements. Currently, several middleware solutions that are expected to facilitate such model are under development.

OpenIoT [1], GSN [2], Cosm [3] are some examples. These middleware solutions strongly focus on connecting sensor devices to software system and related functionalities [1]. However, when more and more sensors get connected to the Internet, the sensor search functionality becomes critical.

This paper addresses the growing challenge of sensor search and selection in IoT solutions and research. Traditional web search approach will not work in IoT sensor selection and search domain as text based search approaches cannot capture the critical characteristics of a sensor accurately. Another approach that can be followed is meta data annotation (e.g. basic details related to each sensor such as sensor type, manufacturer, capability). Even if we maintain meta data on sensors (e.g. stored in sensor's storage) or in the cloud, interoperability will be a significant issue. Further, a user study done by Broring et al. [4] has described an approach where 20 participants were asked to enter metadata for a weather station sensor using a simple user interface. Those 20 persons made 45 mistakes in total. The requirement of re-entering metadata in different places (e.g. enter metadata into GSN once and again enter metadata into OpenIoT) arises when we do not have common descriptions.

Recently, W3C Incubator Group released the Semantic Sensor Network XG Final Report that defines SSNO ontology [5]. SSNO describes sensors, their characteristics and relationships between concepts. SSNO strengthens sensor interoperability and accuracy avoiding error-prone manual data entry. Further, inconsistencies in sensor descriptions can be avoided by letting the sensor hardware manufactures to produce and make available sensor descriptions using ontologies so that IoT solution developers can retrieve and incorporate (e.g. mapping) them into their system. Ontology based sensor description and data modelling is useful for IoT solutions. This approach also allows semantic querying. Our proposed solution allows the users to express their priorities in terms of sensor characteristics and it will search and select appropriate sensors. In our model, both quantitative reasoning and semantic querying techniques are employed to improve the efficiency and performance of the system by utilizing strengths of both techniques.

The rest of this paper is structured as follows: In Section II, we highlight our vision and where the findings of this paper is going to be fit in. We describe CA4IOT (Context Awareness for Internet of Things) architecture in brief in order to emphasize the importance of this research. At the end, we

briefly review relevant literature. Next, we explain problem definitions and motivation in Section III. At the end, we list main research contributions of this paper. In Section IV, we discuss a real world application related to agricultural domain. Our proposed solution, CASSARAM, is presented in detail in Section V. Data models, context frameworks, algorithms, and architectures are also discussed. In Section VI, we provide implementation details including tools, software platforms, hardware platforms, and data sets used in this work. We also discuss assumptions made during the experiments and results of the experiments. Discussion on research findings is presented in Section VII. Finally, we present conclusions and prospects for future work in Section VIII.

II. BACKGROUND AND RELATED WORK

In this section, we briefly explain our broader vision of automated (context-aware) configuration of filtering, fusion and reasoning mechanisms, according to the problems/tasks at hand in IoT paradigm. This will be a significant enhancement over current paradigms, where resources are configured statically and a priori. This vision is part of OpenIoT project [1]. In our previous work, we explained the complete architecture called CA4IOT [6] that facilitates above vision. Here, we briefly introduce CA4IOT and explain how CASSARAM would fit and benefit from CA4IOT vision.

Manually selecting and configuring relevant sensors and data fusion operators when large numbers of sensors are available to use is not feasible [6] or is very hard. For example, users such as environmental scientists (i.e. non-technical personnel) may not have sufficient knowledge in computer science but they need to retrieve sensor data for their work. They are only interested in acquiring relevant data so they can use the data to build models, simulations, understand and solve their problems.

Let us consider a scenario where an environmental scientist wants to measure environmental pollution in Canberra. There is no single sensor that is capable of measuring environmental pollution. For example, environmental pollution can be roughly attributed to three sub categories: land pollution, air pollution, and water pollution. Each category can be measured by a large number of (different types) sensors. Furthermore, there is high level context information that may not become available by processing data retrieved from a single sensor directly. Data retrieved from multiple sensors need to be fused together dynamically at run time to generate such high level context. Manual selection of sensors and data fusion operators in order to facilitate automated sensor configuration could be complex in IoT due to its scale and dynamic nature. Therefore “*How to efficiently select appropriate sensors by understanding the user requirements/problems despite inherent complexity of the challenge?*” is the problem we addressed in CA4IOT. CA4IOT helps the users by automating the task of selecting sensors according to problems/tasks at hand. It focuses on breaking down the user requirements and understanding which sensors can provide relevant information to the users. Once this is completed, CA4IOT needs to find

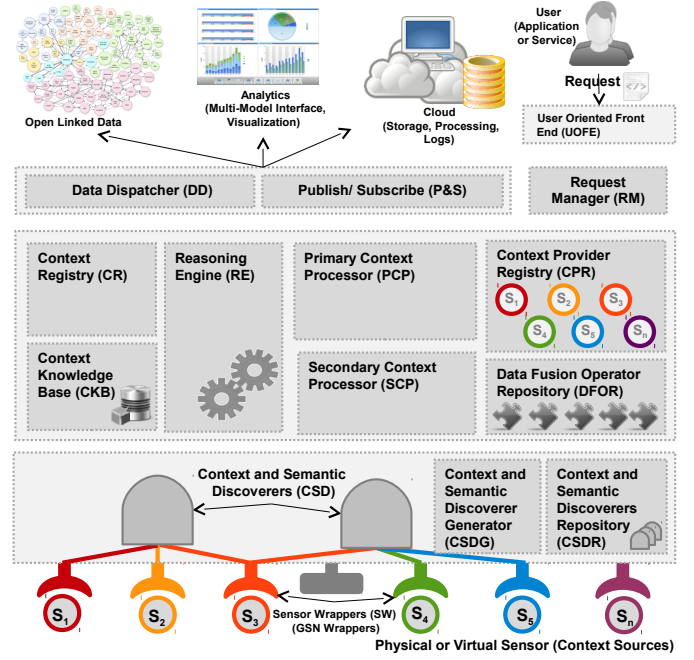


Fig. 1. CA4IOT Architecture supports *sensing as a service* model. Detailed description of this architecture and the execution process is presented in [6].

the most suitable sensors and that is exactly where the thrust of this paper fits in. This problem becomes more challenging when alternatives are available. (e.g. User wants to gather data from 100 temperature sensors located in specific area where 25000 sensors are available while each sensor has different characteristics in terms of accuracy, reliability, cost, etc.). CA4IOT vision is illustrated in Figure 1.

Ideally, IoT middleware solutions should empower users to express what they want and provide the relevant sensor data back to the users quickly without asking the users to manually select sensors which are relevant to their requirements. Even though IoT has received significant attention both in academia and industry, sensor search and selection have not been addressed properly. The following examples show how existing IoT middleware solutions provide sensor searching functionality.

Linked Sensor Middleware (LSM) [7], [8] provides some sensor selection and searching functionality. However, LSM has limited capabilities such as selecting sensors based on location and sensor types. All the searching needs to be done using SPARQL query language which is not very intuitive. Similar to LSM, there are several other IoT middleware related projects under development. GSN [2] is a platform aimed at providing flexible middleware to address the challenges of sensor data integration and distributed query processing. It is a generic data stream processing engine. GSN has gone beyond the traditional sensor network functionality such as routing, data aggregation, and energy optimisation. GSN lists all the available sensors in a combo-box which users need to select. Another approach is Microsoft SensorMap [9]. It only allows users to select sensors by using a location map, by sensor type and by keywords. COSM (formerly Pachube) [3]

TABLE I
COMPARISON OF WEB SERVICES SELECTION AND SENSORS SELECTION DOMAINS

	Web Service Selection Domain	Sensor Selection Domain
Similarities	<ul style="list-style-type: none"> • Consuming single web service may not create significant value. Therefore, web services selection and composition is critical to generate value. • Many alternative web services are available to use • Can be found through directory services • Quality of services matters • There are free as well as paid services 	<ul style="list-style-type: none"> • Collecting data from a single sensor may not create significant value. Therefore, sensor selection and composition is critical to generate value. • Many alternative sensors will be available to use • Middleware solutions such as OpenIoT and GSN will play a mediator roles between sensors and sensor data consumers • Quality of sensors (and data) matters • There will be free as well as paid sensors
Differences	<ul style="list-style-type: none"> • Largely guided by standards • Largely depend on software • Less uncertainty (unless some hardware resources are involved. e.g. weather information.) • Not tangible and more reliable • Some web services accept data as input and produce some data based on them (e.g. data fusion) • Data send to the consumer using web services • Comparatively, less number of web services will be available to access over Internet by 2020 • Typically provide more meaningful processed and refined data. 	<ul style="list-style-type: none"> • No standards (yet) • Largely depend on hardware and firmware • More uncertainty • Tangible, could be mobile and less reliable • Some sensors may accept queries/conditions/preferences as inputs and produce data based on them. Nevertheless, sensors do not accept raw data with the intention of fusing data. • Data will send to the consumer using different techniques such as web services, IP packets, http protocols, etc. • Comparatively, large number of sensors will be available to access over Internet by 2020 • Typically provides less meaningful raw sensor data

is another approach which provides a secure, scalable platform that connects devices and products with applications to provide real-time control and data storage. COSM also offers only keyword search. Our proposed solution CASSARAM can be used to complement the above mentioned IoT middleware solutions with extensive sensor search and selection functionality. Figure 2 shows some of the leading IoT middleware solutions.

In Table I, we present a comparison of similarities and differences between sensor selection and web services selection domains. According to a study in Europe [2], there are over 12,000 working and useful Web services on the Web. Even in such conditions, choice between alternatives (depending on context properties) has become a challenging problem. The similarities strengthen the argument that sensor selection is an important challenge at the same level of complexity as web services. On the other hand, differences show that sensor selection will become a much more complex challenge over the coming decade due to the scale of IoT.

In the following, we briefly describe some of the work done in sensor searching and selection. Truong et al. [12] propose a fuzzy based similarity score comparison sensor search technique to compare output of a given sensor with outputs of several other sensors to find out a matching sensor. Mayer et al. [13] considers location of smart things/sensors as the main context property and structures them in a logical structure. Then, sensors are searched by location using tree search techniques. Search queries are distributively processed in different paths/nodes of the tree. Elahi et al. [14] propose a content-based sensor search approach (i.e. finding a sensor that outputs a given value at the time of a query. Dyser is a search engine proposed by Ostermaier et al. [15] for real-time Internet of Things, which uses statistical models to make predictions about the state of its registered objects (sensors). When a user submits a query, Dyser pulls latest data to identify the actual current state to decide whether it matches the user

query. Prediction models help to find matching sensors with minimum number of sensors data retrievals. Very few related efforts focus on sensor search based on context information.

III. PROBLEM DEFINITION AND MOTIVATION

The problem that we address in this paper can be defined as following. Due to the increasing numbers of available sensors, we need to search and select sensors that provide data which will help to solve our problems at hand in the most efficient and effective way. In order to accomplish this task, we need to search and select sensors based on context. Mainly, we identify two categories of requirements: point-based requirements (non-negotiable or mandatory) and proximity-based (negotiable or flexible) requirements.

First, there are point-based requirements that need to be definitely fulfilled. For example, if a user is interested in measuring temperature in a certain location (e.g. Canberra), the result (e.g. list of sensors) should only contain sensors that can measure temperature. User cannot be satisfied by providing any other types of sensors (e.g. pressure sensors). There is no alternative or compromise in this type of requirements. Location can be identified as a point-based requirement. The second category is proximity-based requirements where users may be willing to suffice with some variations or compromise. For example, user has the same interest as before. However, in this situation, a user may impose proximity-based requirements in addition to the point-based requirements. User may like to have accuracy of the sensors around 92%, and reliability 85%. Therefore, the user gives highest priority to these characteristics. User may accept sensors that closely fulfil his/her requirements even though all other characteristics may not be favourable (e.g. cost of acquisition may be high and sensor response time may be slow). It is important to note that users will not be able to provide any specific value so the system should be able to understand the user priorities and provide the

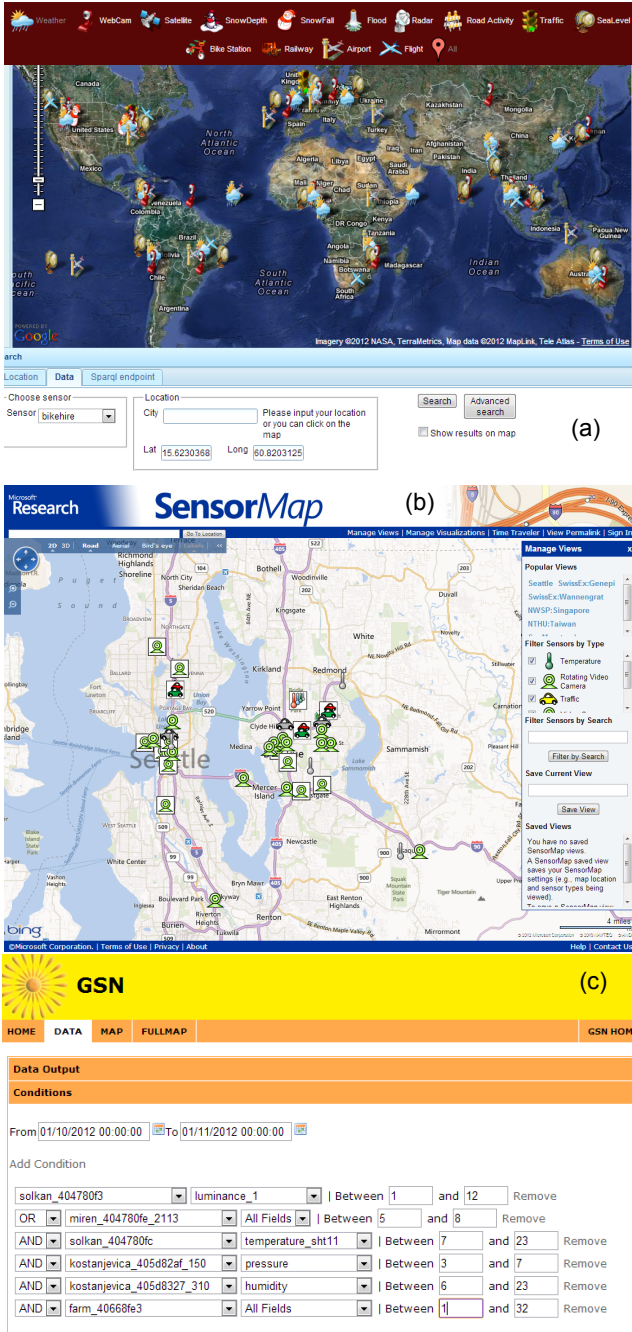


Fig. 2. Some of the leading IoT middleware solutions. Only limited searching functionality is provided across all solutions. (a) Linked Sensor Middleware (LSM) by Digital Enterprise Research Institute [7], [8], (b) SensorMap by Microsoft [9], Global Sensor Network (GSN) initiated by EPFL [2].

results accordingly by using comparison techniques. One of the motivating arguments behind this research work are current market statistics and predictions that show rapid growth in sensor deployments related to IoT and Smart Cities. By 2020, there will be 50 to 100 billion devices connected to the Internet [16]. Further, our work is motivated by increasing trend of IoT middleware solutions development. Today, most of the leading middleware solutions provide only limited sensor search and selection functionality as depicted in Figure 2.

In this paper, we propose a model that can complement any IoT middleware solution. Our contributions can be summarised as follows.

- We propose a context based framework for sensors in IoT middleware which allows to capture and model sensor characteristics. This information allows users to search sensors based on context information.
- We modelled our proposed context framework as an extension to Semantic Sensor Ontology (SSNO) and it is compatible with many existing SSNO-based developments. It is envisioned that the extended SSNO can be used in projects such as Phenonet [17] and OpenIoT [1].
- We propose CASSARAM that allows users to search and select sensors based on user priorities. Users are able to get not only the sensors that will provide required sensor data but also sensors that have characteristics (i.e. context information related to sensors and sensor data acquisition process) that users prefer most. Our approach can be used in Sensing-as-a-Service approach.
- We develop CASSARA tool that allows the users to express their priorities in a comparative manner. Slider UI components allow the comparison. CASSARA also populates a ranked list of sensors in order from best choice to worst.
- We propose a novel technique called *Comparative Priority-based Heuristic Filtering* algorithm to make the sensor indexing algorithms faster and more efficient.
- We evaluate CASSARAM using our prototype tool and measure performance and efficiency in terms of computational resource consumption.

IV. REAL WORLD CHALLENGE

In this section we present a real world example application to reinforce the arguments and to strengthen the necessity of addressing sensor search and selection challenges. It will also help to understand the challenges more clearly. Figure 3 shows state of the art sensor based monitoring system used in Australian agricultural domain. Australia is the fourth largest wheat and barley exporter after US, Canada and EU. There are two challenges that Australian agriculture has to address: scarcity of water resources and low soil fertility. Every year, Australian grain breeders plant up to 1 million 10m² plots across the country to find the best high yielding varieties of wheat and barley. The plots are usually located in remote places often requiring more than four hours travel one-way to reach. The challenge is to monitor the crop performance and growing environment through different seasons and return the information in an easily accessible format. The challenge of crop growing and performance monitoring can be addressed by deploying sensors. Querying the collected sensor data is essential to understand what is happening in the field. The challenge is to develop a sensor-searching model which allows to search sensors based on context information. As we mentioned earlier, it is not required to collect data from all the sensors deployed in all the plots all the time which is inefficient. For example, find out what sensors have more

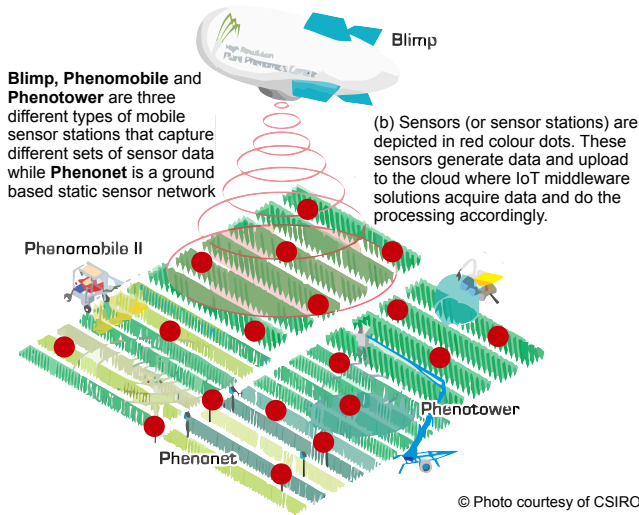


Fig. 3. State of the art sensor based monitoring in agriculture domain.

energy and collect data only from those sensors helps to run the entire network for much longer time without reconfiguring.

V. CASSARAM: THE PROPOSED APPROACH:

In this section, we propose a sensor selection approach step-by-step. First, we provide a high-level overview of the model which describes the overall execution flow and critical steps of the model. Then, we explain how user preferences are captured. Next, data representation model and proposed extensions are presented. Semantic querying and quantitative reasoning techniques are discussed.

The critical steps of CASSARAM are presented in Figure 4. As we mentioned earlier our objective is to allow the users to search and select sensors that best suit their requirements. In our model, we divide user requirements into two categories: *point-based requirements* and *proximity-based requirements*. Point-based requirements are also called non-negotiable requirements and they must be satisfied exactly as specified by the user. In contrast, proximity-based requirements, which are also called negotiable, may or may not be satisfied exactly as specified by the user. Results that are closest to the user requirements would be selected.

Algorithm 1 describes the execution flow of CASSARAM. Common algorithmic notations used in this paper are presented below: *Ontology* (\mathcal{O}) consists of sensor descriptions and context property values related to all sensors, *UserPrioritySet* (\mathcal{P}) contains user priority values for all context properties, *Query* (\mathcal{Q}) consists of point-based requirements expressed in SPARQL, *Number of sensors* (\mathcal{N}) required by the user, *ResultsSet* ($\mathcal{S}_{Results}$) contains selected number of sensors, *IndexedSensorSet* ($\mathcal{S}_{Indexed}$), *Multidimensional Space* (\mathcal{M}) where each context property is represented by a dimension, *UserInput* (\mathcal{UI}) consists of input values provided to CASSARAM by the users via user interface, *ScalingInformation* (\mathcal{SC}) defines the scale of the slider, *WeightedUserPrioritySet* ($\mathcal{P}^{Weighted}$) provides details on how user has prioritised context properties, *ContextPropertySet* (\mathcal{CP}) consists of all context information, *NormalizedContextPropertySet* (\mathcal{NCP}).

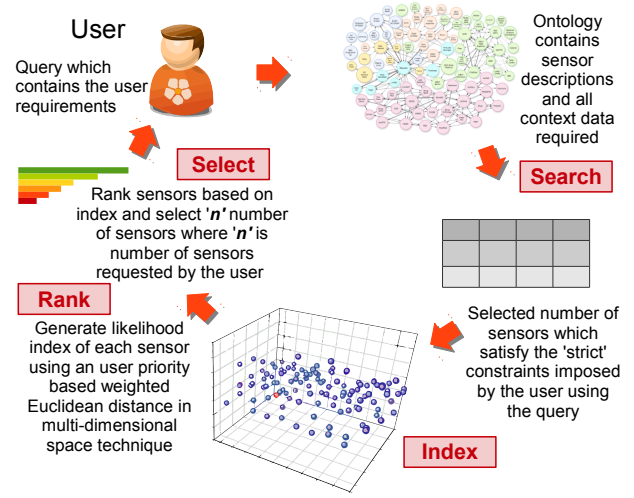


Fig. 4. High level Overview of CASSARAM

At the beginning, CASSARAM identifies point-based requirements, proximity-based requirements, and user priorities. First, users need to select the point-based requirements. For example, a user may want to collect sensor data from 1000 temperature sensors deployed in Canberra. In this situation, sensor type (i.e. temperature), location (i.e. Canberra) and number of sensors required (i.e. 1000) are point-based requirements. Our CASSARA prototype tool provides user interface to express this information via SPARQL queries. SPARQL query is dynamically built when users select the context properties. In CASSARAM, any context property can become a point-based requirement. Then users can define proximity-based requirements. All context properties we discuss in this section are available to be defined in comparative fashion by setting priorities via a slider-based user interface as depicted in Figure 7. Next, each sensor is plotted into a multidimensional space where each dimension is represented by a context property (e.g. accuracy, reliability, latency). Each dimension is normalized 0 to 1. Then, Comparative Priority-based Weighted Index (CPWI) is generated for each sensor combining user priorities and context properties. Finally, sensors are ranked according to CPWI and the number of sensors required by the

Algorithm 1 Execution Flow of CASSARAM

Require: (\mathcal{O}), (\mathcal{P}), (\mathcal{Q}), (\mathcal{N}), ($\mathcal{S}_{Results}$), ($\mathcal{S}_{Indexed}$), (\mathcal{M}).

- 1: **Output:** $\mathcal{S}_{Results}$
- 2: $\mathcal{S}_{Filtered} \leftarrow queryOntology(\mathcal{O}, \mathcal{Q})$
- 3: **if** $cardinality(\mathcal{S}_{Filtered}) < \mathcal{N}$ **then**
- 4: **return** $\mathcal{S}_{Results} \leftarrow \mathcal{S}_{Filtered}$
- 5: **else**
- 6: $\mathcal{P} \leftarrow captureUserPriorities(\mathcal{UI})$
- 7: $\mathcal{M} \leftarrow Plot\ Sensors\ in\ Multidimensional\ Space(\mathcal{S}_{Results})$
- 8: $\mathcal{S}_{Indexed} \leftarrow calculateCPWI(\mathcal{S}_{Results}, \mathcal{M})$
- 9: $\mathcal{S}_{Results} \leftarrow rankSensors(\mathcal{S}_{Indexed})$
- 10: $\mathcal{S}_{Results} \leftarrow selectSensors(\mathcal{S}_{Results}, \mathcal{N})$
- 11: **return** $\mathcal{S}_{Results}$
- 12: **end if**

user are selected from the top of the list.

User priority capturing (UPC) is a technique we developed to capture user priorities through a user interface shown in Figure 5. CASSARAM allows users to express what context property is more important to them compared to others. If a user does not want a specific context property to be considered in the indexing process, they can avoid it by not selecting the check-box correlating to that specific context property. For example, according to Figure 5, *energy* will not be considered when calculating CPWI. This means user is willing to accept sensors with any *energy* consumption level.

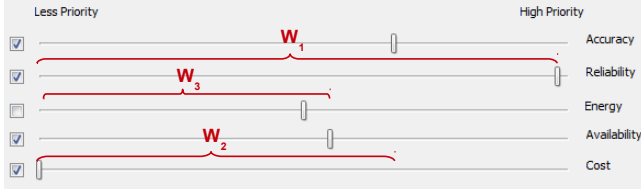


Fig. 5. Weight W_1 is assigned to *reliability* property. Weight W_2 is assigned to *Accuracy* property. Weight W_3 is assigned to *availability* property and finally weight W_4 , default weight, is assigned to *cost* property. High priority means always favourable and low priority means always unfavourable. For example, if user makes *cost* a high priority (more towards right), that means CASSARAM tries to find the sensors that produce data at the lowest cost. Similarly, if user makes *accuracy* a high priority, that means CASSARAM tries to find the sensors that produce data with high accuracy.

As depicted in Figure 5, if users want more reliable sensors to be ranked higher compared to accuracy of the sensors, the reliability slider need to be placed more to the right compared to the accuracy slider. A weight is calculated for each context property. Therefore, more priority means higher weight. As a result, sensors with high reliability and accuracy will be ranked higher. However, those sensors may have high costs due to low priority placed on cost property.

In this paper, we use Semantic Sensor Network Ontology (SSNO) [5] to model sensor descriptions and context properties. Main reasons to select SSNO are interoperability and the trend moving towards ontology usage in IoT and sensor data management domain. A comparison of different semantic sensors ontologies are presented in [18]. The SSNO is capable of modelling significant amounts of information about sensors such as sensor capabilities, performance, the conditions in which sensors can be used, etc. Details are presented in [5]. SSNO includes most common context properties such as accuracy, precision, drift, sensitivity, selectivity, measurement range, detection limit, response time, frequency and latency. SSNO can be extended unlimitedly by sub classing three classes: *measurement property*, *operating property*, and *survival property*. We extend the *quality class* by adding several sub-classes based in order to facilitate our context framework.

In the next step, which we call "Ranking Using Quantitative Reasoning" sensors are ranked based on proximity-based user requirements. We developed a weighted Euclidean distance-based indexing techniques called Comparative Priority-based Weighted Index (CPWI) as follows.

$$(CPWI) = \sqrt{\sum_{i=1}^n [W_i(U_i^d - S_i^\alpha)^2]}$$

First, each sensor is plotted in multidimensional space where each context property is represented by a dimension. Then, users can plot an ideal sensor in the multidimensional space by manually entering context property values as illustrated in Figure 6 by U_i . By default, CASSARAM will automatically plot an ideal sensor as depicted in U_d (i.e. highest value for all context properties). Next, user priorities are retrieved and weights are calculated in comparative fashion.

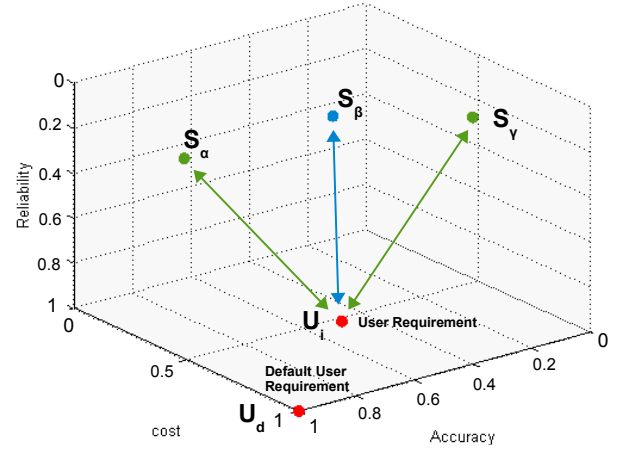


Fig. 6. Sensors plotted in three-dimensional space for demonstration purposes. S_α , S_β , and S_γ represent real sensors. CPWI calculate weighted distance between S_j and $U_i||_d$. Shortest distance means sensor will rank higher because it is close to the user requirement.

We use the following context framework (i.e. context properties listed below) after evaluating a number of research proposals carried out in the areas of quality of service domain related web services [20], mobile computing [21], and sensor ontologies [5], we extracted following context properties to be stored and maintained about each sensor. This information helps to decide which sensor to be used in a given situation. We adopt the following definition to our work. "*Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.*"[22]. CASSARAM has no constraints on a number of context properties that can be used. Our context framework comprises availability, accuracy, reliability, response time, frequency, sensitivity, measurement range, selectivity, precision, latency, drift, resolution, detection limit, operating power range, system (sensor) lifetime, battery life, security, accessibility, robustness, exception handling, interoperability, configurability, user satisfaction rating, capacity, throughput, cost of data transmission, cost of data generation, data ownership cost, bandwidth, and trust. The list is extendible with more context attributes if necessary.

The solution we discussed so far works well with small number of sensors. However, model becomes inefficient when the number of sensors available to search increases. Let us consider an example to identify the inefficiency. Assume we have access to one million sensors. A user wants to select 1,000 sensors out of them. In such situation, CASSARAM will index and rank one million sensors using proximity-

based requirements provided by the user and select top 1,000 sensors. However, indexing and ranking all possible sensors (in this case one million) is inefficient and wastes significant amount of computational resources. Further, CASSARAM will not be able to process large number of user queries due to such inefficiency. We propose a technique called Comparative Priority-based Heuristic Filtering (CPHF) to make CASSARAM more efficient. The basic idea is to remove sensors that are positioned far away from user defined ideal sensor and reduce the number of sensors that need to be indexed and ranked.

Consider the above scenario. First, all the eligible sensors rank in descending order of the highest weighted context property (in this case accuracy). Then, remove 40% (from $N_{Removable}$) of the sensors from the bottom of the list. Then order the remaining sensors in descending order of the next highest weighted context property (in this case reliability). Then, remove 30% (from $N_{Removable}$) of the sensors from the bottom of the list. This process applies for the remaining context properties as well. Finally, index and rank the remain sensors. This approach dramatically reduces the indexing and ranking related inefficiencies. Broadly, this category of techniques are called Top-K selection where top sensors are selected in each iteration.

VI. IMPLEMENTATION AND EXPERIMENTATIONS

The aim of implementation and experimentation is to study the performance of CASSARAM in different IoT related scenarios which we developed based on real world requirements. Experimentation setup, datasets used, assumptions, experiment testbed, and results are presented.

The proposed model analysed and evaluated using a prototype which we developed using Java is called 'CASSARA Tool'. The data was stored in MySQL database. As shown in Figure 7, it allows to capture user preferences regarding their expected priorities on each characteristic of a sensor. We used a computer with Intel(R) Core i5-2557M 1.70GHz CPU and 4GB RAM to evaluate our proposed model. In order to perform mathematical operations such as Euclidean distance calculation in multidimensional space, we used Apache Commons mathematics [23] library. It is an open source optimized library of lightweight, self-contained mathematics and statistics components addressing the most common problems not available in the Java programming language. As we used Semantic Sensor Ontology (SSNO) [5] to manage sensor descriptions and related data, we employed open source Apache Jena API [24] to process and manipulate semantic data. We conducted each experiment 100 times and averages are taken into account.

Our evaluation used a combination of real data and synthetically generated data. We collected environmental linked-data from the bureau of meteorology [25] and data sets from both Phenonet project [17] and Linked Sensor Middleware (LSM) project [7], [8]. The main reasons to combine data are due to the need of generating a large amount of data and the need of controlling different aspects of data (e.g. context

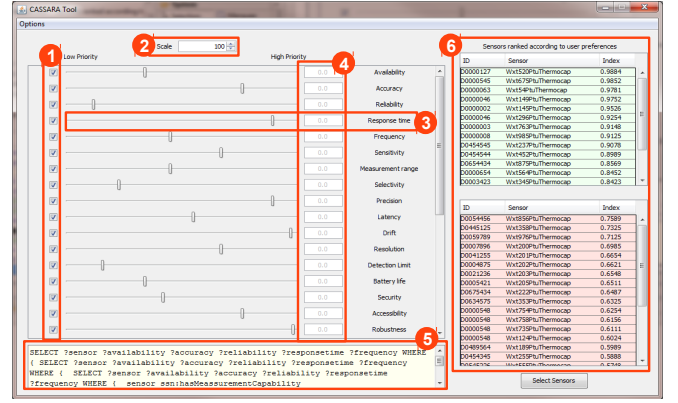


Fig. 7. User interface of the CASSARA tool. (1) Users need to check the box to express that they do concern about that specific context property, (2) Allows to set the scale. Slider becomes more sensitive when scale increased, (3) Slider attach to each context property can be configured to express the priority in comparative fashion, (4) Ideal value related to each context property can be entered. Defaults is zero, (5) Allows to enter SPARQL query that consists point-based requirement. However, by default, tool generates the SPARQL appropriately based on the context properties selected by the users, (6) Shows list of sensors ranked according to index.

information related to sensors need to be embedded into the data set, because real data that matches our context framework is not available in any public data sets at the moment) to better understand the behaviour of CASSARAM in different IoT related real world situations and scenarios where real data is not available.

We make the following assumptions in our work. We assume that sensor descriptions such as sensor capabilities and measurements are already retrieved from sensor manufacturers and merged into SSNO. Similarly, we assume that context data related to each sensor such as current power level, power consumption, accuracy, reliability are retrieved by software systems that manage such data and are available to be used. We acknowledge these data could be stored in a distributed manner (e.g. each GSN instance may contain descriptions of sensors which are connected to that specific instance). Therefore, sensor search may need to be performed in a distributed manner. However, we do not consider distributed aspect in this work and leave it for future work.

In order to evaluate CASSARAM, we used a data set which comprises sensor descriptions and context information for *one million sensors*. We synthetically created this data set by combining different real data sets. In this section, we present experimentation results with brief explanations on each graph. Interpretations of each graph and overall discussions are provided in Section VII.

VII. EVALUATION AND DISCUSSION

As depicted in Figure 8, semantic querying consumes significantly more processing time compared to indexing and ranking. Further, when the number of context properties that are retrieved by a query increases, execution time also increases significantly. MySQL can join only 61 tables which only allows to retrieve maximum of 10 context properties in SSNO data structure. Alternative data storage or running

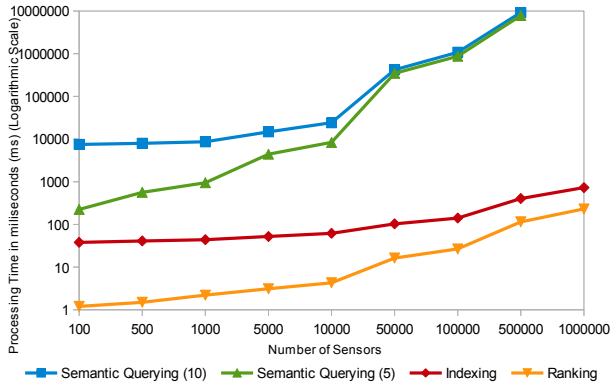


Fig. 8. This graph shows processing time taken by each step and the total sensor selection process when the number of sensors gets increased. The number of context properties used for indexing kept at 30 for ranking and indexing experiences. Semantic querying use 5 and 10 context properties. Note: Y-axis is measured in milliseconds and presented in logarithmic scale.

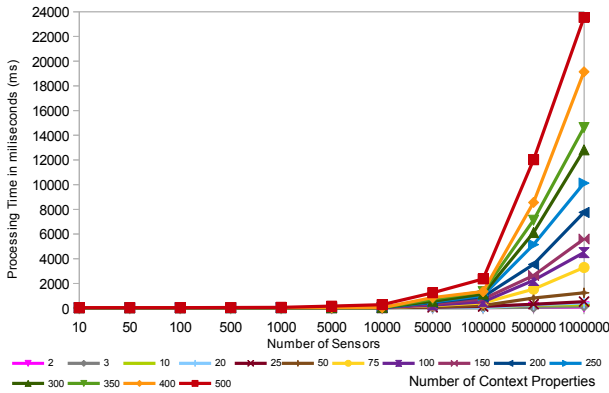


Fig. 9. This graph shows processing time taken by sensor indexing process when number of context properties and number of sensors get increased. Synthetic context properties are used to for experiments.

multiple queries (can be efficient when having fewer sensors. see comparison between semantic querying times) can be used as an alternative solution. We conclude that optimising semantic querying will make significant impact on overall performance of CASSARAM. As depicted in Figure 9, reducing the number of indexed sensors below 10,000 allows to perform CASSARAM faster. Processing time starts to get increased significantly after 100,000 sensors. As depicted in Figure 10, the complexity of CPHF (due to sub queries) has not effected the total processing time significantly. Instead, CPHF saved some amount of time in indexing and ranking phases. CPHF method returns only limited number of sensors where non-CPHF approach returns all the sensors available to CASSARAM which consumes more resources including more processing time, significant amount of memory and temporary storage. According to Figure 11, accuracy of CPHF approach increases when margin of error (M) increases. However, lower M leads CASSRAM towards low resource consumption. Therefore, it is a trade-off between accuracy and resource consumption. The optimum value of M can be dynamically learned by machine learning techniques based on what context properties are prioritized by the users in each situation and how the normalized weights are distributed across different

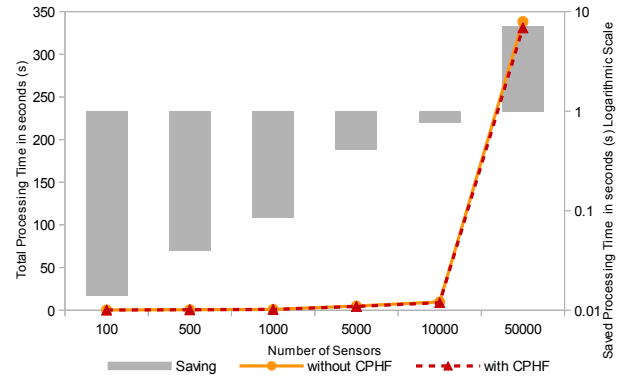


Fig. 10. This graph compares time taken by sensor selection process with and without CPHF algorithm when number of sensors get increased. Number of sensors that users require kept at 50 in all experiments ($N=50$). 30 context properties are retrieved via semantic querying, indexed and ranked.

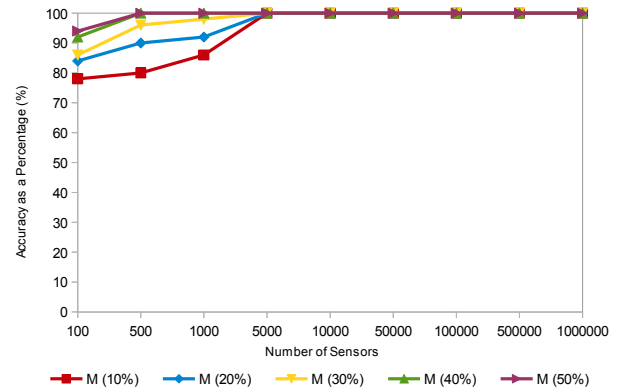


Fig. 11. This graph shows how accuracy changes when Margin of Error ($M\%$) value changes in CPHF algorithm and number of sensors get increased. Number of sensors that users require kept at 50 in all experiments ($N=50$).

context properties. We will investigate these possibilities in future research.

VIII. CONCLUSIONS AND FUTURE WORK

With the advances in sensor hardware technology and cheap materials, sensors are expected to be embedded into many objects around us which increases the number of sensors available to us. This means that we have access to multiple sensors that would measure similar environmental phenomenon. We need to decide what operational and conceptual sensor related context properties are more important than others.

In this work, we showed how context information related to each sensor can be used to search and select sensors that are best suited for user requirements. We selected sensors based on user expectations and priorities. As proof of concept, we built a working prototype to demonstrate the functionality of our CASSARAM and to support experimentations in realistic applications. We also highlight how CASSARAM will help us to achieve our broader Sensing-as-a-Service vision in IoT paradigm. In future, we plan to incorporate CASSARAM into leading IoT middleware solutions such as GSN, SensMA, and OpenIoT to support automated sensor selection functionality in distributed environment. This will help us to perform more evaluations and understand how automated sensor selection would complement IoT middleware solutions. Further, we will

investigate how semantic and quantitative reasoning can work together more closely to achieve efficient results and to provide more functionality.

IX. ACKNOWLEDGEMENT

Authors acknowledge support from SSN TCP, CSIRO, Australia and ICT OpenIoT Project (Open source blueprint for large scale selforganizing cloud environments for IoT applications), which is co-funded by the European Commission under seventh framework program, contract number FP7-ICT-2011-7-287305-OpenIoT. The Author(s) acknowledge help and contributions from all partners of the OpenIoT project.

REFERENCES

- [1] OpenIoT Consortium, "Open source solution for the internet of things into the cloud," January 2012, <http://www.openiot.eu> [Accessed on: 2012-04-08].
- [2] GSN Team, "Global sensor networks project," 2011, <http://sourceforge.net/apps/trac/gsn/> [Accessed on: 2011-12-16].
- [3] Cosm, "Cosm platform," 2007, <https://cosm.com/> [Accessed on: 2012-08-05].
- [4] A. Bröring, F. Bache, T. Bartoschek, and C. P. Elzackker, "The sid creator: A visual approach for integrating sensors with the sensor web," in *Advancing Geoinformation Science for a Changing World*, ser. Lecture Notes in Geoinformation and Cartography, S. Geertman, W. Reinhardt, and F. Toppen, Eds. Springer Berlin Heidelberg, 2011, pp. 143–162. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-19789-5_8
- [5] w3.org, "Semantic sensor network xg final report: W3c incubator group report," June 2011, <http://www.w3.org/2005/Incubator/ssn/XGR-ssn-20110628/> [Accessed on: 2012-09-25].
- [6] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Ca4iot: Context awareness for internet of things," in *IEEE International Conference on Green Computing and Communications, Conference on Internet of Things, and Conference on Cyber, Physical and Social Computing (iThing)*, Besancon, France, November 2012.
- [7] Digital Enterprise Research Institute, "Linked sensor middleware (lsm)," 2011, <http://lsm.deri.ie/> [Accessed on: 2012-09-24].
- [8] D. L. Phuoc, H. N. M. Quoc, J. X. Parreira, and M. Hauswirth, "The linked sensor middleware - connecting the real world and the semantic web," in *International Semantic Web Conference (ISWC)*, October 2011.
- [9] S. Nath, J. Liu, and F. Zhao, "Sensormap for wide-area sensor webs," *Computer*, vol. 40, no. 7, pp. 90–93, Jul. 2007. [Online]. Available: <http://dx.doi.org/10.1109/MC.2007.250>
- [10] Cisco Blog, "The internet of things," The Internet of Things 2011, <http://blogs.cisco.com/news/the-internet-of-things-infographic/> [Accessed on: 2011-08-08].
- [11] P. Norvig, "Teach yourself programming in ten years," 2001, <http://norvig.com/21-days.html> [Accessed on: 2012-04-16].
- [12] C. Truong, K. Romer, and K. Chen, "Fuzzy-based sensor search in the web of things," in *Proceedings of the 3rd International Conference on the Internet of Things (IoT 2012)*, Wuxi, China, 2012.
- [13] S. Mayer, D. Guinard, and V. Trifa, "Searching in a web-based infrastructure for smart things," in *Proceedings of the 3rd International Conference on the Internet of Things (IoT 2012)*, Wuxi, China, 2012.
- [14] B. M. Elahi, K. Romer, B. Ostermaier, M. Fahrmaier, and W. Kellerer, "Sensor ranking: A primitive for efficient content-based sensor search," in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*, ser. IPSN '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 217–228. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1602165.1602186>
- [15] B. Ostermaier, K. Roalter, L. O. andmer, F. Mattern, M. Fahrmaier, and W. Kellerer, "A real-time search engine for the web of things," in *Proceedings of the 2nd International Conference on the Internet of Things (IoT 2010)*, 29 2010-dec. 1 2010, pp. 1–8.
- [16] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelffle, "Vision and challenges for realising the internet of things," European Commission Information Society and Media, Tech. Rep., March 2010, http://www.internet-of-things-research.eu/pdf/IoT_Clusterbook_March_2010.pdf [Accessed on: 2011-10-10].
- [17] Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia, "Phenonet: Distributed sensor network for phenomics supported by high resolution plant phenomics centre, csiro ict centre, and csiro sensor and sensor networks tcp," 2011, <http://phenonet.com> [Accessed on: 2012-04-20].
- [18] M. Compton, C. Henson, H. Neuhaus, L. Lefort, and A. Sheth, "A survey of the semantic specification of sensors," in *2nd International Workshop on Semantic Sensor Networks, at 8th International Semantic Web Conference*, Oct. 2009. [Online]. Available: <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-522/p6.pdf>
- [19] C. Bizer and A. Schultz, "The berlin sparql benchmark," *Int. J. Semantic Web Inf. Syst.*, vol. 5, no. 2, pp. 1–24, 2009.
- [20] S. Ran, "A model for web services discovery with qos," *SIGecom Exch.*, vol. 4, no. 1, pp. 1–10, Mar. 2003. [Online]. Available: <http://doi.acm.org/10.1145/844357.844360>
- [21] D. Chalmers and M. Sloman, "A survey of quality of service in mobile computing environments," *Communications Surveys Tutorials, IEEE*, vol. 2, no. 2, pp. 2–10, quarter 1999.
- [22] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, ser. HUC '99. London, UK: Springer-Verlag, 1999, pp. 304–307. [Online]. Available: <http://dl.acm.org/citation.cfm?id=647985.743843>
- [23] Apache Foundation, "Commons math: The apache commons mathematics library," 2011, <http://commons.apache.org/math/> [2012-09-05].
- [24] Apache Software Foundation, "Apache Jena," November 2010, <http://jena.apache.org/> [accessed on: 2012-05-10].
- [25] Australian Government, Bureau of Meteorology, "Experimental environmental linked-data published by the bureau of meteorology," 2012, <http://lab.environment.data.gov.au/> [Accessed on: 2012-009-05].