WoT Search Engine based on Multi Agent System: A Conceptual Framework

https://doi.org/10.3991/ijim.v16i05.27901

Seif Eddine Mili[1](✉), Vincent Rodin[2]
[1]Ecole Normale Supérieure, Constantine, Engineering Laboratory for Complex Systems (LISCO), Annaba University, Algeria
[2]University of Brest, Brest, France
seifmili@gmail.com

Abstract—The rising incursions with embed and linked sensors cause the construction of the Internet of Things (IoT). It's presenting a wide access to the reality. We support that on head of this data, a WoT is required. A framework that permits the achievement of programs related to sensors in the IoT. In this paper we present a conceptual model that describes Web of Things Search engine (WoTSE). Our WoTSE is based on artificial multi agent system with a modular architecture, addressing the challenge of scalable search for fast altering content while benefiting on the existing internet structure. Furthermore, we propose a conceptual search framework for the Web of Things, "WoTmAS2E", with experimental evaluation using a prototype.

Keywords—IoT, WoT, search engine, multi agent system

1 Introduction

The present world is being more and more encrusted by sensors which can be linked to the Internet and the Web allowing to observe more and more things almost in real time with a browser. Increasing everyday objects like appliances, cars and equipment with embedded computer or visible markers permits matters and records approximately them to be on hand numerically (via the internet, cellular, etc.) [1].

IoT is a technological realization of omnipresent computing where technology is included normally into everyday objects. Especially interesting, this idea opens the way to many scenarios based on the connection between the reality world and the virtual world: home automation, health, smart city, logistics... [2]. However, like other promising concepts, this one faces a number of technical and non-technical challenge that need to be investigated in order for the IoT to reach its full potential [2].

With the reuse of technology and strategies of the internet navigation, the data and utility of those things like sensors diffusion, may be supplied at the Web as sources for customers and applications [3].

We think that this vogue is ancestor of a WoT [4] which will develop the classic Web, providing a standard link between reality entities and the virtual. This will be accessible via software applications. The entities representations include all state perceived by sensors (occupied parking place, empty parking place cool bar). However, in the IoT, sensors and actuators which might be linked to Smart Things display and manipulate their adjacent location. They produce crude data about state changes in the environment, which does not interest the users. Taking a case, if we are looking for sensors with a instant information between 25° and 32°, we suppose that users will be attracted the search for a room that has an ambient temperature.

A smart agent may be a piece of software program which is reactive, proactive, perceptible and social. It has capacity to take a look at and act upon an environment [5]. A multi-agent system is a system composed of more than one interacting smart agents [6]. Using multi-agent structure to implement personalization mechanism for Web of things Search Engine (WoTSE) has great advantages. User hobbies alternate overtime, agent has the capacity to understand the alternate of user hobbies actively and update in time. Moreover, it is additionally vital to merge results and offer recommendation more flexibly in line with search context. In summary, agent improves the adaptability of WoTSE.

The main contribution of this paper is proposing a conceptual model that describes WoTSE based on artificial multi agent system with a modular architecture addressing the challenge of scalable search for fast altering content while benefiting on the existing internet structure. We used Multi Agent System (MAS) method to increase the precision of information retrieval and evaluation criteria by creating agents to respond to the different personalization issues and phases [17].

The MAS approach is used here to improve the accuracy of data search and take into account valuation standards thru constructing a couple of agents to deal with specific issues and customization steps.

Basically, our framework like [1] takes into consideration instant user queries to deliver a result inspired by real-world entities.

This paper is structured as: Section 2, related work is review. Section 3 and 4 report the proposed multi agent system Web of Things multi Agent System Search Engine (WoTmAS2E) architecture. In Section 5, we describe our implementation efforts. In section 6 we develop experimental evaluation. Finally we conclude our work in Section 7.

2 Related work

WoTSE are cataloger of Web of Thing [3]. They find out and collect Web of Thing resources in a selected field and permit customers to find on that source [3].

We distingue five utilization forms and implementation of WoTSE in literature that are summarizes in the following table:

Table 1. Description of different WoTSE using scenario

Utilization Forms	Description	Example of Systems Included
Locating Physical	WoTSE are typically used to find objects, which might be tagged with passive radio frequency identification (RFID) tags	MAX, Object Calling Home
Sensor Search	Using WoTSE for retrieving sensors totally based on their information, like profitability	GSN, CASSARAM
Finding Actuation Services	Using WoTSE like a middle ware for retrieving offerings provided through objects	Ref [13]
Retrieving Data Records	The usage of WoTSE to retrieve information applicable to object	EPC Discovery Services, Cooltown [14]
Finding Entity with Dynamic State	Using WoTSE for real time search for real world object	Dyser

We describe in Table 1 the different forms of use and implementation of WoTSE and for each category we have briefly mentioned some systems that belong to it. We detail below the different systems shown in Table 1.

MAX is an autonomous method that permits customers to offer a multitude of expressive words to discover their labeled material items. To replicate a real individual behavior, MAX results the area of the corresponding items as mark in preference to coordinates. Requests are broadcasted via the net from the bottom hosts, and the identification of the radio frequency detect corresponding items are referred as mark. [3].

Object Calling Home System is a system who makes use of its taking part cellular as a sensor net to identify lost objects. Every item is connected with a battery-powered frequency Bluetooth diffuser, that's found through the phone's integrated Bluetooth discovery mechanism. [7, 8, 9].

Global Sensor Network system (GSN) is a platform for incorporating WIFI net sensor on the Internet and not an engine for search. But, it is referenced a lot because of its sensor option system when treatment sensor diffusion. Global Sensor Network types every sensor in the network as virtual [3]. When processing sensor diffusion consistent with the deployment description of consumers, GSN makes use of this data to retrieve sensors and carry out the treatment [10].

CASSARAM is a system which is looking for connected sensors the usage of their contextual data, including reaction time, reliability, accuracy and availability [3]. CASSARAM use (SSNO) a semantic ontology to explain the data domain [11]. The user must use this ontology via a special tool (SPARQL) which generates a result through the CASSARAM graphical interface. Best ranked sensors are returned as a result search [12].

Benoit Christophe et al. [13] develop system that proceeds like a module in a wide Web of Things application structure rather than an autonomous system. To illustrate entities, [13] use more than one ontology to describe their output and their input structures, end state machinery, owners and emplacements. All entities are classified by the analogy among their input constitution and the output constitution of the requested entities.

EPC Discovery Services who try to construct a quality evaluation framework for conducting an evaluation of the existing Discovery Services architectures [3].

Dyser is a search engine based on content for the WoT. It operates like traditional search engines as yahoo or other, wherein the Dyser index is constructed on keywords, also customers use easy and constructed request language to look for entity. A search engine of Dyser is in real time for any Entities. In, [1] requests concerning states of Entities are treated at the moment of the request. The system of Dyser do not index old information, it marks web page keywords and prediction models to be able to predict future states of Entities [1].

3 WoTmAS2E model conceptual view

Since WoT and IoT infrastructures are real time and dynamic, the processing time of the request may be greater than the frequency of changes of state of the object, this means that the response to the request becomes inexact and out of date. To face this situation we must use prediction models to anticipate the next state of the object and reduce the search time. This allows the indices to be always up to date.

As shown in Figure 1, the user interacts with the Web of Things Multi Agent System Search Engine "WoTmAS2E" model via introducing a request, which is then adjusted to provide an adjusted request. The request is adjusted through the Entities of Interest "the main query is composed of one or many Entities and Entity is composed of one or many sensors". After that, the adjusted request is sent to be compared with external environment "internet" for fetching the results of the search and assembled to generate customized results. The WoTmAS2E finally output the results using the click method to maintaining the resource collections up to date.

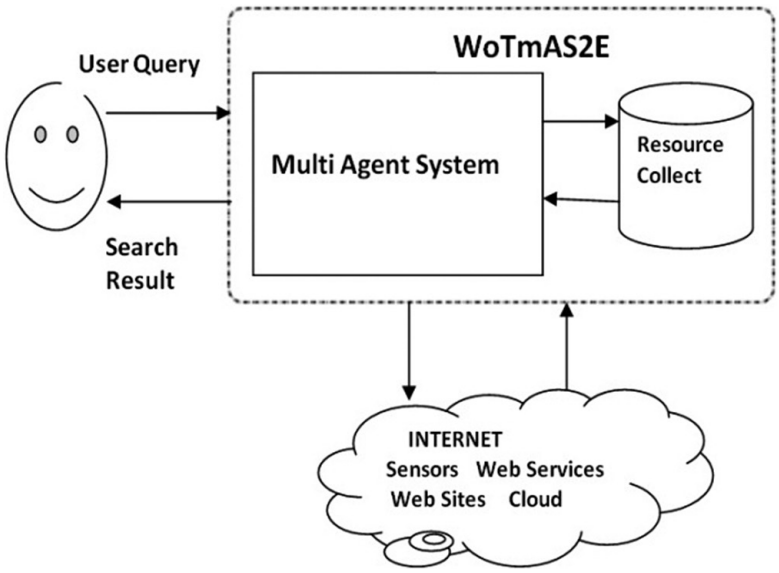

Fig. 1. The WoTmAS2E model conceptual view

4 WoTmAS2E model architecture

Our Web of thinks Search engine model composition use the multi agent system method that may be simply enlarged and conserved. The agents inside the MAS are completely asynchronous this is one of the most characteristics of multi agent system, which means the agents communicated and worked in parallel. As shown in Figure 2, WoTSE includes 4 agents: User Agent, index Agent, manage Agent and crawler Agent.

Obtaining the results of the search, the four agents interchanges through ports by using a communication language, KQML for Knowledge Query Manipulation Language. To provide the search request and retrieve the results, users must communicate with user agent via interface. In order to adjust the request of the user to be transmitted to the crawler agent, the manage agent take in to consideration the Entities of Interest provided by main query "user agent". Once the crawler agent receives the adjusted request, it interacts with external environment such as sensors, web services, to recover and broadcast the results of the search, and after it returns them to the resource collections. The result are sent to the manage agent to be aggregate and then sent to the index agent to be ranked with the prediction model. The personalized final results of the search are given to user by the user agent.

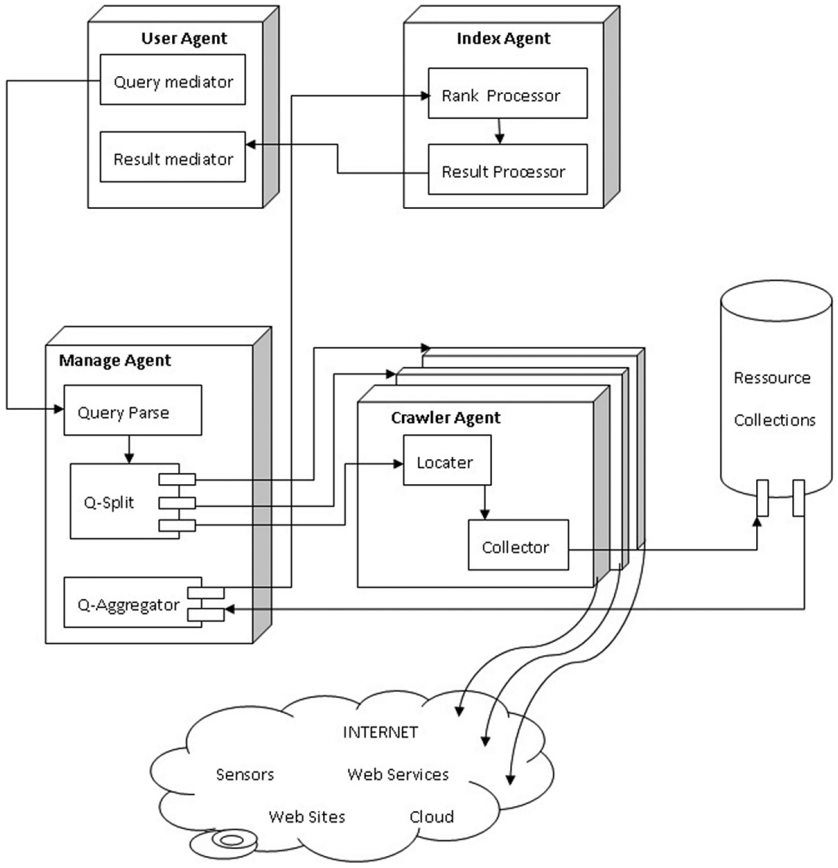

Fig. 2. The WoTmAS2E model architecture

4.1 User agent

The User Agent is a mediator agent among the customer and the system. It gives the customer with a graphical user interface for creating search queries and for presenting the user search results. Then, the user Agent has two major components: Query mediator and Results mediator

Query mediator: to receive queries, it is in charge for communication with the user to obtain the user request.

Result mediator: return search results, it is in charge for giving the final results of the search to the customer.

4.2 Manage agent

The manage Agent has a reactive feature and is in charge for the management and communication between the different agents of the system. This agent communicates with the user agent through the query parse module and through query split with the crawler agent and finally through Query Aggregator with index agent. The manage Agent forwards the adjusted search request user to a number of crawler agents and after it aggregates the results into one list after verification of each and every downloaded Web page against the concerned user query, this one is ready to be remit to the index agent. Manage Agent has three main components: Query Parse, Query Split and Query Aggregator.

Query Parse: This module remold brute customer request into a structure usable by the system, ready to be splitted.

Query Split: this module is responsible for splitting the query to multiple sub query that can be deal separately by crawler agent.

Query Aggregator: this module is in charge for combining different Resource Collections results into a final score for each query, to send it to index agent for ranking.

4.3 Crawler agent

The crawler Agent has a reactive feature and is in charge for detecting resources in the WoT and collects this one to be stored in resource collections. For every sub query coming from manage agent (after parsing and splitting) we do an instantiation of crawler agent. The crawler Agent has two main components: locater and collector.

Locater: this module detects resources detailed in the search engine and the relation among resources and objects.

Collector: this element amasses resources discovered and moves them in the Resource Collections.

4.4 Index agent

The index Agent has a reactive feature and is in charge for indexing and ranking the downloading results from crawler agents, the results downloaded from crawler agents are saved in resource collections and used by manage agent to be aggregate and sent to

the index agent. It should be delivered to the user. After successful verification by manage agent (query aggregator), the Web pages are further ranked using a ranked analysis. Then, these ranked Web pages are handed over to the user Agent which in-turn sends the acknowledged documents to the user. The index Agent has two main components: Rank Processor and Result Processor.

Rank Processor: We use a specific ranker in this module to rank resources according to their normal classification, regardless of the user's queries.

Result Processor: take out the data from matching result resources and provide search results.

4.5 Resource collections

The Resource Collections is a database where agents store information and data.

5 Implementation

In this section, we describe the implementation of our prototype of WoTmAS2E using JAVA, Web Services Applications and KQML for communication among agents. The most important steps of the search engine are: sensors representation, crawling, indexing, and ranking. We show below the implementation of these components.

5.1 Sensor and entity of interest representation

We used the same prototypical micro format like Dyser [1]. We outline the shape of sensor web page in HTML element, the information inside the element are an identification number that identify sensor, GPSloc gives details concerning the position of sensor, sensor type is specified by Kind, and present State lists the status that the sensor is presently detecting and possible States consists of a listing of states. Figure 3 shows the HTML description of sensor micro-format.

For the creation of the entity of interest page, we do like Dyser [1], we include a link to sensor page and we include a specific attribute which states that they are a link between sensor page and entity of interest page. The crawler agents analyze HTML pages and follow the attached links, make use of this data to locate entity pages and their related sensors.

```
▼<div class="sensor">
   <div class="idfn"> sensdb21</div>
  ▼<div class="GPSloc">
     <div class="long">45.8</div>
     <div class="lat"> 13.5</div>
   </div>
   <div class="kind"> temperature</div>
   <div class="presentstate"> hot </div>
  ▼<div class="PossibleState">
     <div class="State"> hot </div>
     <div class="State"> cold </div>
     <div class="State"> medium </div>
   </div>
   <div class="typeofperdictionmodel"> MultiplePeriodPredictionModel </div>
  ▼<div class="perdictionmodel">
     <!---JSON serialized version of the prediction
     Model -- -->
   </div>
 </div>
```

Fig. 3. HTML description of sensor micro-format

5.2 Crawling

An algorithm of the sensor method life cycle is shown below. The sensor gateway with push method, index the information send by sensor. We then contacts sensor gateways directly, by the use of service, to get the Uniform Resources Locator of sensors and entities of interest. The results are then analyzed, loaded and the contents are saved in the resource collections. Gateway hears at these own sensors and up to date states depending on sensor values. The update states for model prediction are based on a quantity of successive state changes. The extraction of the type of prediction model like multi period prediction model [15] is not considered in the context of this paper's scope.

```
Algorithm 1: sensor gateway process life-cycle
Input: time T
Output: sensor value
Step 1 read value from sensor
Step 2 if current value <> previous value then send
current value, goto step 4
Step 3 if current value = previous value then goto step 4
Step 4 every regular interval T got step 1
End.
```

5.3 Indexing

In our prototype, we implement the index with a relational database, which interpreted by JDBC interface. This permit a wide selection of multiple database

implementations. We use the MYSQL database with InnoDB and stores data as documents with JavaScript Object Notation format.

We use Xively to permit to the users connecting and publishing sensor evaluation on the network, and make their own Internet of Things applications by mixing these sensors, sensor evaluation, and network information, using the Xively Application. The Uniform Resource Locator of the Xively Application is http:api.xively.com, from which it is possible to securely read and write sensor information, read and write metadata, and read ancient sensor information via HTTP methods like GET, POST, and PUT.

Outputs of every indexed data are modeled inside the database for a given length of time for which forecasts are to be prepared. Therefore, there are no data in the database, but the discretization of the output model prediction data for a definite length of time for which forecasts are to be prepared. Searching the entities with top probability of corresponding the outputs of sensor is therefore executed like a database search operation.

5.4 Rank process

The probability that a sensor S is in some status x is SP(x), for example, occupied. PT(t,x) is the probability to be in certain state x in time t, t is a periodic time (week, hour month…). F is the factor of certainty that a sensor can be in a state x, where Fϵ[0,1]. We have

$$\sum_{x=1}^{n} F(x) = 1 \tag{1}$$

Where n is the total number of state. Thus, SP(x) is evaluated as follows:

$$SP(x) = PT(t,x) * F(x) \tag{2}$$

An entity is defined by one or more sensors, we name State Entity EP(x) as the probability that entity is in status x. Since the EP(x) can be based on one or many sensor status, EP(x) is calculated as follows:

$$EP(x) = \sum_{y=1}^{m} SP(y) * IF(y) \tag{3}$$

Where m is the quantity of sensors that entity status is based on, SP (y) is the personal sensor states and IF(y) is the factor impact of sensor y in EP(x).

It is the entity state that defines the ranking process.

6 Evaluation

We assume that, there are Web of Things networks that supervise many smart establishments. In our evaluation, we supposed that a user would like a calm and big meeting area in the establishments. According to this situation, we will use our search engine to fulfill the requested needs of the user.

The WoTmAS2E evaluation was performed on Xively. The research will be carried out on the indices selected by the WoTmAS2E giving a ranked listing of predicted Entities of Interest which must be estimated.

WoTmAS2E was rated on indices that are makes by datasets and random values. We apply a pragmatic datasets who's used by the gateway of sensor as an alternative of true sensors. User defines a sensor type and decides how many establishments are concerned and quantity of devices in each one.

We used virtual sensors with the size of the forecast interval size set to seven days (one week), when we realized the search engine.

The most important WoTmAS2E evaluation criteria are: index size, execution time (i.e., time elapsed for different steps of searching), and resulting execution.

Table 2. The WoTmAS2E sizes index and times execution for all execution run steps of quantity of things and establishments

Week Model	7 Days	Devices	Index	Results	Time_run(ms)
EXE_run_stp	Establishments				
1	500	50	87500	3845	52
2	1000	100	350000	19570	185
3	1000	200	700000	40237	320
4	50000	200	35000000	1885446	11371
5	50000	200	35000000	125	806

Table 2 gives index sizes, establishments and the quantity of devices for every establishment at different execution run step, by which every execution run step is different compared to quantity of establishments and devices for every establishment. The product of the quantity of establishments, the size of the forecasting horizon and half the quantity of devices for every establishment defines the index size. Compared to the other WoTSE (Dyser) our index size does not increase considerably if the number of sensors increases.

Our system builds its indices, based on The Resource Collections datasets that manage the crawl processing, via the index and downloads of particular data about the Web of Things. As a consequence, the number of recurrent processing crawl was diminished with the time elapsed for indexing and analyzing Web of Things pages. For example the developers of Dyser [15] used an additional service created by Mayer, [8] for analyzing many formats by which Web of Things networks data are designed, therefore it monopolize time.

The time elapsed for the search in the WoTmAS2E is based on the quantity of establishments and the quantity of devices in each establishment. The WoTmAS2E provide to the users the capacity to select the number of buildings and the number of devices per building to diminish network overload.

Figure 4 presents the relationship among the process time at the different execution run step listed in Table 2. The time consumed by our system increases or decreases depending on the size of the index and crawl processing.

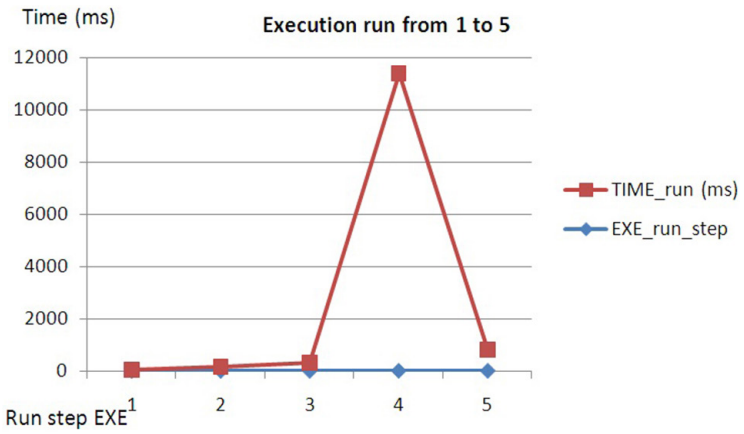

Fig. 4. Time processing of WoTmAS2E evaluation

Our multi agent system architecture feature rise the capacity of retaining indices updated, whereby manage agents sends only aggregated information to the index agents.

We attest that WoTmAS2E profit from the distributed techniques of the multi agent system for query processing. Therefore, this reduces the time consumed for indexing, analyzing, and crawl processing.

7 Conclusion

Users prefer searching for top level states of entities [16]. In this paper, we preset a multi agent system designed for searching on the WoT via a Search Engine. This system assists the user to refine the preferred search results. This system is consisted of four different kinds of agents by which linked to each other with particular logic.

We presented the structure and prototype implementation of multi agent web of things search engine, a real time search engine which allows to find real entity which presents a definite status when the request occurs.

We evaluated the achievement of WoTmAS2E based on a realistic data via datasets and random synthesized values.

In WoTmAS2E, distributed agents deal with the crawl processing and index processing, in order to provide combined data to the mange agent, while an index preserves précis data accuracy about the search engine. WoTmAS2Edecreases the instant of the crawl processing using t he distributed and parallel execution of agents.

8 References

[1] Römer, K., Ostermaier, B., Mattern, F., Fahrmair, M., & Kellerer, W. (2010). Real-time search for real-world entities: A survey. *Proceedings of the IEEE, 98*(11), 1887–1902. https://doi.org/10.1109/JPROC.2010.2062470

[2] Benjamin, B. (2015). Data Stream Management System for the Future Internet of Things. these de doctorat de l'Université de Versailles Saint-Quentin-En-Yvelines, France. https://tel.archives-ouvertes.fr/tel-01166047

[3] Tran, N. K., Sheng, Q. Z., Babar, M. A., & Yao, L. (2017). Searching the web of things: State of the art, challenges, and solutions. *ACM Computing Surveys (CSUR), 50*(4), 1–34. https://doi.org/10.1145/3092695

[4] Guinard, D., Trifa, V. M., & Wilde, E. (2010). Architecting a mashable open world wide web of things. Technical Report/ETH Zurich, *Department of Computer Science*, 663.

[5] Vafadar, S., & Barfourosh, A. (2015). Towards Intelligence Engineering in Agent-Based Systems. (IAJIT).

[6] Wang, M., Li, Q., Lin, Y., Li, Y., & Zhou, B. Y. (2018). A personalized metasearch engine based on multi-agent system. In SEKE (pp. 707–706). https://doi.org/10.18293/SEKE2018-082

[7] Frank, C., Bolliger, P., Mattern, F., & Kellerer, W. (2008). The sensor internet at work: Locating everyday items using mobile phones. *Pervasive and Mobile Computing*, 4(3), 421–447. https://doi.org/10.1016/j.pmcj.2007.12.002

[8] Mayer, S., Guinard, D., & Trifa, V. (2012, October). Searching in a web-based infrastructure for smart things. In 2012 3rd IEEE International Conference on the Internet of Things (pp. 119–126). https://doi.org/10.1109/IOT.2012.6402313

[9] Wang, H., Tan, C. C., & Li, Q. (2009). Snoogle: A search engine for pervasive environments. *IEEE Transactions on Parallel and Distributed Systems, 21*(8), 1188–1202. https://doi.org/10.1109/TPDS.2009.145

[10] Aberer, K., Hauswirth, M., & Salehi, A. (2007, May). Infrastructure for data processing in large-scale interconnected sensor networks. In 2007 International Conference on Mobile Data Management (pp. 198–205). IEEE. https://doi.org/10.1109/MDM.2007.36

[11] Compton, M., Barnaghi, P., Bermudez, L., Garcia-Castro, R., Corcho, O., Cox, S., ... & Taylor, K. (2012). The SSN ontology of the W3C semantic sensor network incubator group. *Journal of Web Semantics*, 17, 25–32. https://doi.org/10.1016/j.websem.2012.05.003

[12] Perera, C., Zaslavsky, A., Christen, P., Compton, M., & Georgakopoulos, D. (2013, June). Context-aware sensor search, selection and ranking model for internet of things middleware. In 2013 IEEE 14th international conference on mobile data management (Vol. 1, pp. 314–322). IEEE. https://doi.org/10.1109/MDM.2013.46

[13] Christophe, B., Verdot, V., & Toubiana, V. (2011, September). Searching the'web of things'. In 2011 IEEE Fifth International Conference on Semantic Computing (pp. 308–315). IEEE. https://doi.org/10.1109/ICSC.2011.69

[14] Kindberg, T., Barton, J., Morgan, J., Becker, G., Caswell, D., Debaty, P., ... & Spasojevic, M. (2002). People, places, things: Web presence for the real world. *Mobile Networks and Applications*, *7*(5), 365–376. https://doi.org/10.1023/A:1016591616731

[15] Ostermaier, B., Römer, K., Mattern, F., Fahrmair, M., & Kellerer, W. (2010, November). A real-time search engine for the web of things. In 2010 Internet of Things (IOT) (pp. 1–8). IEEE. https://doi.org/10.1109/IOT.2010.5678450

[16] Younan, M., Khattab, S., & Bahgat, R. (2016, May). Wotsf: a framework for searching in the web of things. In Proceedings of the 10th International Conference on Informatics and Systems (pp. 278–285). https://doi.org/10.1145/2908446.2908496

[17] Moawad, I. F., Talha, H., Hosny, E., & Hashim, M. (2012). Agent-based web search personalization approach using dynamic user profile. *Egyptian Informatics Journal, 13*(3), 191–198. https://doi.org/10.1016/j.eij.2012.09.002

9 Authors

Seif Eddine Mili is an assistant professor in the Department of Computer Science, Ecole Normale supérieure Constantine, Algeria. He holds a PhD in Computer science from Badji Mokhtar Annaba University, Algeria. His research interest includes Web of Things, IoT, Multi Agent System, Bio inspired System and Model Driven Architecture.

Vincent Rodin is Professor in Computer Science. He holds a PhD in computer science from INP of Toulouse, France and a HDR (Research accreditation) in computer sciences from University of Rennes I, France. His research activities focus mainly on the use of Multi-Agent Systems for the modelling and the simulation of biological and environmental processes. Currently, he has Full Professor position at University of Brest, Lab-STICC/CNRS 6285, France (Email: vincent.rodin@univ-brest.fr).

Article submitted 2021-10-27. Resubmitted 2022-01-08. Final acceptance 2022-01-08. Final version published as submitted by the authors.