Organizing Capabilities using Formal Concept Analysis

Wassim Derguech, Souleiman Hasan, Sami Bhiri, Edward Curry
National University of Ireland, Galway - Digital Enterprise Research Institute
{*wassim.derguech, souleiman.hasan, sami.bhiri, ed.curry*}*@deri.org*

***Abstract*—In this paper, we discuss the importance of the concept of capability for describing what an action does from a functional perspective. We introduce a conceptual model for representing capabilities as attribute features entities. Furthermore, we use Formal Concept Analysis to create concept lattices for indexing a repository of capabilities descriptions. We applied this approach on a set of sensor capabilities deployed within the Linked Energy Intelligence dataspace.**

***Keywords*-Formal Concept Analysis; Capability Modelling; Discovery; Sensor Capability**

I. Introduction

Capabilities represent the actions performed or the information delivered by a component such as a service, a business process, a sensor, a computer program, etc. A proper capability description allows users to discover the component that satisfies a particular need. A rich text that describes what the component does is a good way to properly address this issue. However, reading a full text for determining the capability of a component is not always convenient especially in the context of interactions between machines. Here, it is required to have a proper capability description that serves both machine processing and human understanding.

In this context and with the advent of the Semantic Web, ontologies and languages introduced several ways to describe service capabilities [1], [2], [3], [4]. However, we argue that this was not properly addressed. First, current approaches consider capabilities as annotated invocation interfaces and not as functionalities. Second, capabilities are described in terms of Inputs, Outputs, Preconditions and Effects which requires from the user to read the documentation of the service in order to determine what it does. Third, current approaches provide little information on how services with similar capabilities are related.

Additionally, with the rapid growth of services, sensors, smart devices, programs, etc. what is required is a well defined organizing strategy of capabilities descriptions. A proper indexing of capabilities helps to provide an efficient discovery mechanism. Actually, OWL-S[1] specification provides a mechanisms for defining hierarchies of service descriptions called *profile hierarchies*. This is useful for classification purposes but it does not allow to find services that even if they do not share the same set of properties still provide the same functionality. Feng et al. [5] propose an indexing structure for linking capabilities while analyzing their attributes relations and propose a heuristic approach for the discovery phase. However, a heuristic approach does not guarantee an optimal solution.

The research problem that we are dealing with in this paper is twofold. First, capabilities are not properly described for serving both machine processing and human understanding. Second, current capability (or service description) repositories are not efficiently organized.

As a solution to this problem, we propose to define capabilities as stand alone entities without any ties to their implementations. Indeed, we propose the concept of *Structured Entity* in Section II for modeling any resource that can be defined via a set of attributes. We use this concept to model sensor capabilities as a set of domain specific properties. These properties include an action verb and a set of attributes that are defined in a domain related ontology providing the possible values each attribute can have [6]. As a solution for organizing capabilities, we propose to use Formal Concept Analysis (FCA). FCA is a well-known mathematical classification tool used in various domains that allows organizing objects described via a set of attributes into a *Concept Lattice*. We introduce in Section III the theoretical foundations of FCA where we introduce main concepts used along this paper. Then in Section IV we introduce Linked Energy Intelligence dataspace which constitutes our use case for organizing sensor capabilities using FCA. Finally, Section VI draws conclusion and future work after discussing and relating our work with existing approaches in Section V.

II. Modelling Capabilities as Structured Entities

A capability denotes what an action does either in terms of world effects or returned information[2]. In the literature, we can distinguish three families of approaches that tackled the problem of capability modeling either directly or indirectly. The first family includes Semantic Web Services models (WSMO[3] and OWL-S[4]) which model capabilities as Input, Output, Preconditions and Effects (IOPE paradigm).

[1]OWL-s: http://www.w3.org/Submission/OWL-S/

[2]OASIS Reference Model for Service Oriented Architecture 1.0, http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf

[3]WSMO: http://www.wsmo.org/

[4]OWL-s: http://www.w3.org/Submission/OWL-S/

Modeling capabilities as such does not feature in an explicit and easily accessible way domain features which makes their management difficult and heavily dependent on reasoning.

The second family of related efforts concerns semantic annotations of invocation interfaces (SA-WSDL[5] and SA-REST[6]). While these approaches do not directly target capability modeling, they attempt to provide alternative solutions to top-down semantic approaches (WSMO[3] and OWL-S[4]) by starting from existing descriptions such as WSDL[7] and annotate them with semantic information. These approaches define a semantic description of syntactic interaction interfaces rather than concrete capabilities.

The third family includes frame-based approaches for modeling capabilities. Oaks et al. [1] give a comprehensive overview of related approaches and propose a model for describing service capabilities as such. The proposed model distinguishes in particular the corresponding action verb and informational attributes (called roles in the paper [1]) in addition to the classical IOPE. While this model makes a step beyond the classical IOPE paradigm, the semantics of capabilities remain defined via the IOPE paradigm and therefore has the same issues as the first family of approaches described above.

All of the previously discussed approaches describe capabilities without featuring functional domain properties. A capability is highly coupled with its implementation (i.e, invocation interface) or related to the description of another concept (i.e., services). We strongly support the idea of considering the capability as an independent concept that describes what a program, a business process, a service, etc. does from a functional perspective. A capability should not be limited only to a single label or an action verb but also should consider a proper description of functional domain related properties forming a standalone *structured entity*.

We present a *structured entity* as a general concept that can serve any conceptualization for modeling any attribute featured entities. Particularly, in our case, we apply it for modelling capabilities. A structured entity is any object/thing which is described via a set of attributes. An attribute is a property (in RDF terms) but it is a special kind of property; it is an intrinsic property of the entity. For instance, *name*, *date of birth*, *address*, *friend-of*, *work-in* are all properties of an entity *Person*. However, the former three properties are intrinsic properties of a person while the later two express its relationships to other entities (another person and another organization). Definition 1 introduces the concept of a structured entity.

Definition 1 (Structured Entity): A structured entity SE is a set of pairs *(AttributeName, AttributeValue)*. An *AttributeName* corresponds to the identifier of the attribute and *AttributeValue* corresponds to the value of the entity in question.

In our work, we adopt this approach of modeling any structured entity by a set of attributes and apply it in modeling capabilities [6]. For example, if we intend to model a sensor capability we need to describe its functional properties such as what that sensor does, whether or not the sensor has a digital display, etc. Listing 1 presents an example of a temperature sensor capability *:TemperatureSensorCapability123* that has as attributes *cap:hasActionVerb* with the value *sco:Sensing* and *sco:hasDigitalDisplay* with the value *:LCDDisplay*.

```
@prefix sco: <http://.../sensor_capability_ontology#>.
@prefix cap: <http://vocab.deri.ie/cap#>.

:TemperatureSensorCapability123 a cap:Capability.

:TemperatureSensorCapability123 cap:hasActionVerb sco:Sensing.

:TemperatureSensorCapability123 sco:hasDigitalDisplay :LCDDisplay.
```

Listing 1. Snippet of the Temperature Sensor Capability

The attribute *cap:hasActionVerb* is a special mandatory attribute. This attribute has been previously introduced in [1] in order to define, in a natural language, what action is performed by a particular service. We define action verbs with respect to an action verb schema [8] where we define the concept *ActionVerb* as an *rdfs:subClassOf skos:Concept*. The proposed schema defines for this concept two properties: *av:hasPart* and av:hasOptionalPart which are used to build a hierarchy of action verbs expressing meronymy relations between them. A meronymy relation holds between two concepts if one of them is a part of the other. The MIT process handbook[9], NAICS[10] and UNSPSC[11] are good examples of actions ontologies/taxonomies that can be reused for the action verb attribute. As in this work, we are interested in modeling sensor capabilities, all the capabilities will have the same action verb which is *sco:Sensing*.

In previous work [6], we presented in details our conceptual model for capability description. In this paper, we propose an indexing structure that organizes and facilitates the efficient discovery of capabilities modeled according to [6] in a particular repository. To this end, we propose an approach based on Formal Concept Analysis.

III. Formal Concept Analysis for organizing Capabilities

The approach that we adopt in this work consists of using Formal Concept Analysis [7] (FCA for short) for better organizing a repository of capabilities in order to make their discovery more efficient. We define in this paper the

[5]SA-WSDL: http://www.w3.org/2002/ws/sawsdl/
[6]SA-REST: http://www.w3.org/Submission/SA-REST/
[7]WSDL: http://www.w3.org/TR/wsdl
[8]http://vocab.deri.ie/av
[9]MIT Process Hanbook: http://process.mit.edu/
[10]NAICS: http://www.census.gov/eos/www/naics/
[11]UNSPSC: http://www.unspsc.org/

theoretical foundations of FCA while applying it on sensors capabilities.

FCA is a technique that has evolved from the mathematical lattice theory and used for data analysis across several domains. Examples of domains include organizing web search results into concept based on common topics, gene expression data analysis, information retrieval, understanding and analysis of source codes, etc. [8]. It represents a powerful tool for identifying meaningful relationships within a set of objects that share common attributes. It provides as well a theoretical model to build from a *formal context* (see Definition 2) a partially ordered structure called a *concept lattice*.

Definition 2 (Formal Context): A *formal context* $\mathcal{FC}$ is a triplet $< X, Y, I >$ where X and Y are non-empty sets and $I \subseteq X * Y$ is a binary relation between X and Y.

For a formal context $\mathcal{FC}$, elements $x \in X$ are referred to as objects and elements $y \in Y$ are called attributes. $< x, y > \in I$ denotes that the object x has the attribute y.

In our work, the formal context is actually defined via the set of sensors that we have as well as their respective descriptions. We will use Table I (called cross-table) as a running example which describes the relationship between the objects (i.e., sensors 1 to 5 represented by the table rows) and their descriptions (i.e., attributes represented by the table columns). Each entry in the table containing X indicates that the corresponding sensor has the corresponding attribute. We consider in this example four attributes: *Active* that indicates if the sensor is in operation; *Storage Option* that indicates if the sensor has the possibility to store data on it; *Digital Display* that indicates if the sensor is equipped by a digital display for displaying the data and finally *Accessible* that indicates if the sensor is located in an accessible area. In our example we have X ={Sensor 1, Sensor 2, Sensor 3, Sensor 4, Sensor 5}, Y ={Active, Storage Option, Digital Display, Accessible}, and Table I reports on I.

Table I
DATA TABLE WITH BINARY ATTRIBUTES

	Active	Storage Option	Digital Display	Accessible
Sensor 1	X	X	X	X
Sensor 2	X		X	X
Sensor 3		X	X	X
Sensor 4		X	X	X
Sensor 5	X			

Another fundamental concept in FCA is the *Formal Concept*. This concept is defined in Definition 3.

Definition 3 (Formal Concept): A *formal concept* in $< X, Y, I >$ is a pair $< E, I >$ of $E \subseteq X$ (called *extent*) and $I \subseteq Y$ (called intent) such that $Att(E) = I$ and $Obj(I) = E$.

$Att(E)$ is an operator that assigns subsets of X to subsets of Y, such that $Att(E)$ is the set of all attributes shared by all objects from E. $Obj(I)$ is an operator that assigns subsets of Y to subsets of X, such that $Obj(I)$ is the set of all objects sharing all the attributes from I.

From Definition 3, we can conclude that a concept $C =< E, I >$ is created by getting objects from E sharing the same attributes from I. For example, the shaded rectangle in Table I represents a formal concept $< E_1, I_1 >$=<{Sensor 1, Sensor 2, Sensor 3, Sensor 4}, {Digital Display, Accessible}> because $Att(E_1) = \{DigitalDisplay, Accessible\}$ and $Obj(I_1)$={Sensor 1, Sensor 2, Sensor 3, Sensor 4}.

From a *formal context* $\mathcal{FC} =< X, Y, I >$ we can deduce a set of formal concepts that can be ordered with respect to a subconcept ordering. Definition 4 formally introduces the subconcept ordering.

Definition 4 (Subconcept Ordering): Having two formal concepts $< E_1, I_1 >$ and $< E_2, I_2 >$ from $\mathcal{FC} =< X, Y, I >$, $< E_1, I_1 > \leq < E_2, I_2 > \iff E_1 \subseteq E_2$ ($\iff I_2 \subseteq I_1$) .

Let's consider the following formal concepts from the example of Table I:

$< E_1, I_1 >$=<{Sensor 1, Sensor 2, Sensor 3, Sensor 4}, {Digital Display, Accessible}>

$< E_2, I_2 >$=<{Sensor 1, Sensor 2, Sensor 4}, {Digital Display, Accessible}>

$< E_3, I_3 >$=<{Sensor 1, Sensor 2}, {Active, Digital Display, Accessible}>

$< E_4, I_4 >$=<{Sensor 1, Sensor 2, Sensor 5}, {Active}>

Then

$< E_3, I_3 > \leq < E_1, I_1 >$, $< E_3, I_3 > \leq < E_2, I_2 >$,

$< E_3, I_3 > \geq < E_4, I_4 >$ and $< E_2, I_2 > \leq < E_1, I_1 >$.

The set of ordered formal concepts derived from a formal context is called a *concept lattice* which is another important notion in FCA. A concept lattice can be represented into a graph such as the one depicted in Figure 1[12]. In this figure, the concept extent near the bottom of the lattice contains only *Sensor 1* since the corresponding intent is related to the biggest number of attributes. The top concept contains all the sensors and its intent corresponds to no attribute. This makes the concept less interesting as it allows for all possibles combinations of attributes.

So far, we considered binary attributes (i.e., either the object has or not that attribute). However in real settings when describing capabilities, there are also multi-valued attributes. Consider Table II, this table contains an additional attribute *Observed Phenomenon*. This attributes reports whether the sensor is an *Energy* consumption sensor, *Light* detection sensor, *Temperature* sensor or a *Motion* sensor. In this case, we need to transform this multi-valued attribute into a binary attribute.

For the usage of FCA, transforming and preprocessing the data Table II is needed. One of the possible ways

[12]All concept lattices in this paper are created using Conexp. [http://conexp.sourceforge.net/]

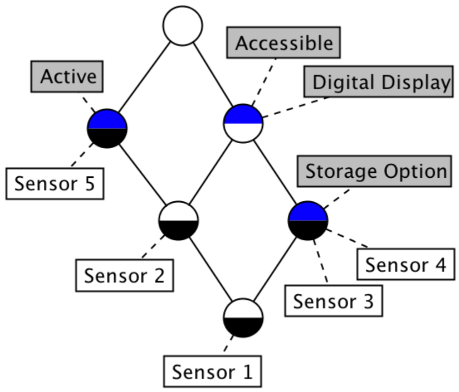

Figure 1. Concept Lattice of the example of Table I.

Table II
DATA TABLE WITH A MULTI-VALUED ATTRIBUTE

	Active	Storage Option	Digital Display	Accessible	Phenomenon Observed
Sensor 1	X	X	X	X	Energy
Sensor 2	X		X	X	Energy
Sensor 3		X	X	X	Light
Sensor 4		X	X	X	Temperature
Sensor 5	X				Motion

consists of using scaling method. Scaling is a transformation method that converts a multi-valued attribute into a context. Table III[13] represents the transformation of the multi-valued attribute *Phenomenon Observed* into a context.

Table III
DATA TABLE WITH A SCALED MULTI-VALUED ATTRIBUTE

	PO: Energy	PO: Light	PO: Temperature	PO: Motion
Sensor 1	X			
Sensor 2	X			
Sensor 3		X		
Sensor 4			X	
Sensor 5				X

After the application of FCA on converted tables the resulted lattice is depicted on Figure 2.

IV. USE CASE APPLICATION

We illustrate in this section a use case scenario using a set of real world sensors deployed within the Linked Energy Intelligence (LEI) dataspace. LEI is an ecosystem where energy related data is made available and interlinked to support decision making and ultimately energy consumption friendly behaviour [9]. Such data is provided by real-time data sources such as sensors as well as relatively static background knowledge such as building plan and occupancy. The LEI dataspace has been realized in the Digital Enterprise Research Institute (DERI).

[13]Please note that PO stands for Phenomenon Observed

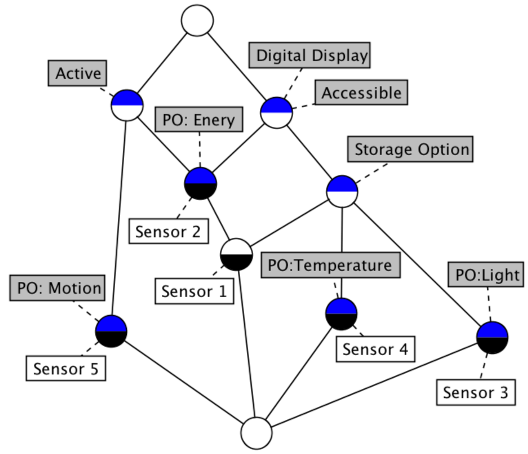

Figure 2. Concept Lattice of the example of Table II.

DERI is a premier research institute with approximately 130 research students and staff with a worldwide reputation in its area. It is based in a dedicated building with 2190 sq. m of space, comprising 22 unit offices, 160 open plan workspaces, 1 large 80-seat conference room with audio visual and video conferencing facilities, 4 meet-ing rooms, 3 kitchens, 1 air conditioned data centre with backup generator, 1 sensor network laboratory, a 30 person café, and Ireland's National Museum of Computing History.

There are various sources of power consumption in DERI such as Heating, Ventilation and Air Conditioning (HVAC) systems, lights, and electronic devices. The building provides a first-class technical infrastructure to its researchers. In addition, the University Computer Services Group provides significant network support to DERI and the technical administrative team within DERI provides the institute's computer infrastructure technical support.

The DERI building has been retrofitted with energy sensors to monitor the consumption of power within the building. In total there are over 50 fixed energy consumption sensors covering office space, café, data centre, kitchens, conference and meeting rooms, computing museum along with over 20 mobile sensors for devices, light and heaters energy consumption as well as light, temperature and motion detection sensors. A building-specific aspect of the dataspace has been presented in [10] with a sensor network-based situation awareness scenario presented in [11]. For this work we ended up with a total number of 78 sensors.

These sensors are described via a set of attributes:

- Active: This attribute reports whether the sensor is in operation.
- Observed Phenomenon: we have four observed phenomenon which are *"energy and power consumption"*, *"motion"*, *"light"* and *"temperature"*. This attribute is a multivalued attribute that needs to be scaled using the transformation previously shown on Table III.

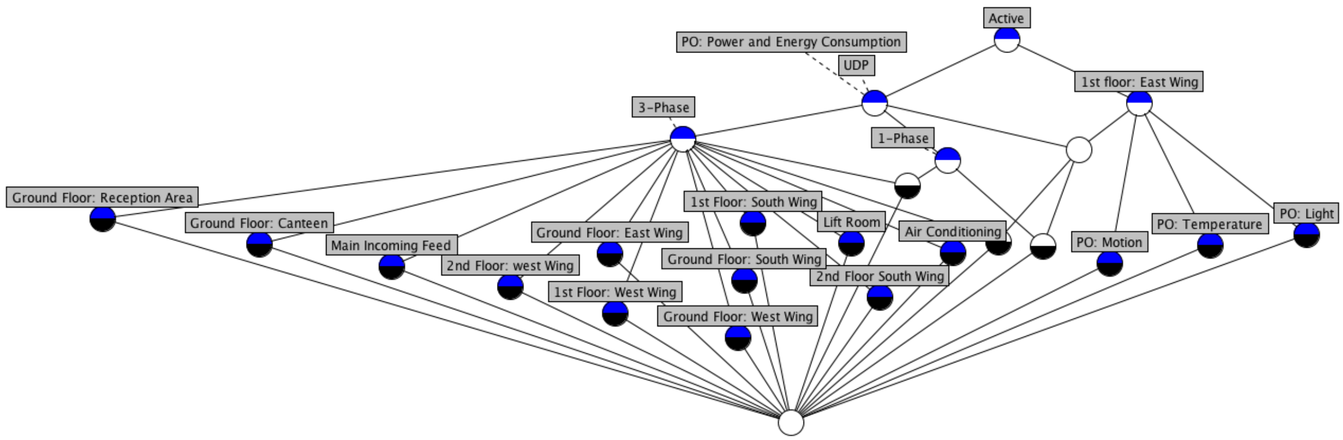

Figure 3. Concept Lattice of the LEI-DERI use case.

- Protocol: This attribute indicates the protocol used by the sensor. We have in our selection of sensors two possible protocols: UDP used by electricity and power consumption sensors and CoAP used by other sensors. This is a multi-valued attribute that has to be scaled.
- Electricity Phases: This attribute reports on the electricity phases used by the sensor, we have in our use case two options: 3-phases and 1-phase sensors. Again this is a multi-valued attribute that has to be scaled.
- Location: even though this attribute is not an intrinsic property of the sensor, we have used it because it is an important information that is required for processing the data provided by the sensor. This is also a multi-valued attribute that enumerates the locations of the sensors, e.g., 1st floor: west wing, ground floor: canteen, main incoming feed, etc. that needs to be scaled.

The resulting concept lattice when applying FCA on our dataset is depicted in Figure 3. The top concept in this lattice represents the set of all active sensors <{Sensor 1, Sensor 2, ... Sensor 78}, {Active}>. This formal concept contains in its extend all the sensors of our dataset because they are all active. We can see in this concept lattice several formal concepts that represent the set of motion sensors <{Sensor 61,... Sensor 66}, {OP: Motion}>, the formal concept for temperature sensors <{Sensor 67,... Sensor 72}, {OP: Temperature}> and the light sensors <{Sensor 73,... Sensor 78}, {OP: Light}>. These three formal concepts are all subconcepts of the concept <{Sensor 61, ... Sensor 78}, {1st Floor:East Wing}>. This helps to deduce that all motion, temperature and light sensors are installed in the same location, i.e., 1st Floor : East Wing.

One of the main advantages of using FCA is the construction of formal concepts which extent (i.e., the set of objects) represents an *equivalence class* of objects sharing the same attributes. This equivalence class can be seen as an abstraction of its set of objects. This is useful in case of replacement of one sensor by another that shares similar characteristics. For example, if one of the motion sensors is not active anymore, it is directly possible to use one of the other motion sensors in his equivalence class. This reduces the change time [14] considerably to simply parsing the lattice until reaching the required equivalence class and select one of its sensors rather than performing a full search over the set of all the available sensors.

The other advantage of this approach is the presence of the explicit relationship between equivalence classes. This relationship is represented via the subconcept relation between formal concepts. This helps optimize the discovery of a particular sensor. The search request becomes a simple path in the concept lattice starting from the top concept. This also helps avoid having empty results. In fact, during the navigation of the concept lattice, if the user cannot find the equivalence class that satisfies his request, he can adapt his request according to the visited nodes of the lattice. This allows the user to relax his query by reducing the attributes he initially identified in his request.

V. Related Work

We discussed in Section II relevant contributions to service descriptions. However, in this section we discuss works related to similarity based discovery of services.

Similarity-based service discovery has been studied extensively for various kinds of services. In [12], multiple evidences are considered to evaluate similarity for service operations and inputs/outputs based on *clustered concepts* extracted from WSDL documents. In [13], a hybrid service matchmaking is proposed for OWL-S services. Among the hybrid filters presented in [13], the *subsumed-by* and *nearest-neighbor* filter leverage different Information Retrieval (IR) similarity metrics. In [14], a *replacement degree* is computed

[14]Change time: the required time for selecting a replacement sensors for the disabled one.

for two *service protocols* based on how their sub-protocols can replace each other in the context of mediated service interactions. Comparing to the above approaches, we are more focused on providing an easy way for the users to specify their requirements to find a replacement for an unsatisfiable service capability by providing the set of required attributes (i.e., service properties).

Feng K. et al. [5] also describe sensor services using our approach (attribute-featured description of services). However, they provide an indexing structure different from FCA concept lattice. It relies mainly on differences between the descriptions that are captured via *extension* and *specification* relations. Using these relations, they are able to build a tree used in a heuristic searching algorithm. Using FCA is more appropriate for avoiding a heuristic search as the indexing structure (i.e., concept lattice) is uniformly created. A search algorithm requires only to navigate that lattice.

VI. Conclusion

The concept of capability is an important element that describes what an action does from a functional perspective. It is important to model this concept in order to allow for a better discovery for a particular service, business process, application, etc. We propose in our work to model capabilities as attribute-featured entities where each attribute reports on a particular characteristic of the described action. Our conceptual model is flexible enough to consider even non-functional attributes to include for instance quality of service attributes. We applied this approach for modeling sensor capabilities, from a real world dataspace, featuring both functional and non-functional attributes. For example we used the attribute location of the sensor even though it was not a functional attribute. On top of these sensor descriptions, we used Formal Concept Analysis (FCA) for indexing these sensors. The resulting indexing structure is called concept lattice that can serve for several use cases for example the discovery of a replacement sensor. Using FCA in such use case is recommended only if the number of objects (i.e., sensors) is not very big. Actually, this constitute the major disadvantage of our approach.

As part of our future work, we plan to provide the required algorithms for updating this indexing structure (i.e., removing or adding a sensor description). In fact, when introducing a new sensor description, we should reconstruct the concept lattice from the beginning. This could not be very useful in highly dynamic environments when several sensors could be added and removed frequently or where the values of attributes might change after a certain period. We plan also to investigate more the scaling operation in the case of multi-valued attribute especially when attribute have complex types such as conditions, dynamic values, etc.

Acknowledgment

This work is funded by the Lion II project supported by Science Foundation Ireland under grant number 08/CE/I1380.

References

[1] P. Oaks, A. H. M. ter Hofstede, and D. Edmond, “Capabilities: Describing what services can do,” in *ICSOC*, ser. LNCS, vol. 2910. Springer, 2003.

[2] J. Kopecký, E. P. B. Simperl, and D. Fensel, “Semantic Web Service Offer Discovery,” in *SMRR*, ser. CEUR Workshop Proceedings, vol. 243, 2007.

[3] T. Vitvar, M. Zaremba, and M. Moran, “Dynamic Service Discovery Through Meta-interactions with Service Providers,” in *ESWC*, ser. LNCS, vol. 4519. Springer, 2007.

[4] M. Zaremba, T. Vitvar, M. Moran, and T. Haselwanter, “WSMX Discovery for the SWS Challenge,” in *ISWC*, Athens, Georgia, USA, November 2006.

[5] F. Gao, W. Derguech, and S. Bhiri, “Ubiquitous Service Capability Modeling and Similarity Based Searching,” in *ESE*, 2012.

[6] S. Bhiri, W. Derguech, and M. Zaremba, “Modelling capabilities as attribute-featured entities,” in *WEBIST (Selected Papers)*, ser. LNBIP, vol. 140. Springer, 2012.

[7] B. Ganter and R. Wille, *Formal Concept Analysis - Mathematical Foundations*. Springer, 1999.

[8] R. Belohlavek, “Introduction to Formal Concept Analysis,” 2008.

[9] E. Curry, S. Hasan, and S. O’Riáin, “Enterprise Energy Management using a Linked Dataspace for Energy Intelligence,” in *SustainIT*. IEEE, 2012.

[10] E. Curry, J. O’Donnell, E. Corry, S. Hasan, M. Keane, and S. O’Riáin, “Linking Building Data in the Cloud: Integrating Cross-Domain Building Data using Linked Data,” *Advanced Engineering Informatics*, 2013.

[11] S. Hasan, E. Curry, M. Banduk, and S. O’Riáin, “Toward Situation Awareness for the Semantic Sensor Web: Complex Event Processing with Dynamic Linked Data Enrichment,” in *SSN11*, Bonn, Germany, 2011.

[12] X. Dong, A. Y. Halevy, J. Madhavan, E. Nemes, and J. Zhang, “Similarity Search for Web Services,” in *VLDB*. Morgan Kaufmann, 2004.

[13] M. Klusch, B. Fries, and K. P. Sycara, “Automated semantic web service discovery with OWLS-MX,” in *AAMAS*. ACM, 2006.

[14] Z. Zhou, F. Gao, and L. Shu, “Service Protocol Replaceability Assessment in Mediated Service Interactions,” in *ICC*. IEEE, 2011.