ASeCS: Assistive Self-Care Software Architectures for Delivering Service in Care Homes

Reza Shojanoori, Radmila Juric, Mahi Lohi, Gabor Terstyanszky
Faculty of Science and Technology, University of Westminster, London, UK

Abstract

We propose a layered and component based software architecture, which generates semantic software applications, for the purpose of delivering personalized services for residents in Self-Care Homes (SeCH). The architectural core layers accommodate software components which grasp and understand the semantic of various situations we may encounter in SeCH, through a variety of cyber-physical objects which co-exist in pervasive environments used in monitoring SeCH residents. The decision making on appropriate actions in SeCH is based on reasoning created by SWRL enabled OWL ontologies to ensure that in any situation, residents are delivered suitable and personalized healthcare services. The ASeCS architecture has been deployed through component based Java technologies, and uses OWL-API in order to seamlessly incorporate reasoning into software applications. ASeCS is SeCH specific, but provides a window of opportunities for creating modern and flexible software solutions for pervasive healthcare, where decision making solely depends on OWL/SWRL enabled computations.

1. Introduction

The healthcare domain gives some of the most successful examples of where the application of modern software technologies, advances in mobile and wireless environments and the power of pervasive computing has materialized [1][2][3][4][6]. The issue of having enormous number of devices with variable communication and computational power embedded into our everyday life has become almost common in healthcare. In support of the technological advances, new software solutions also have been developed which support the delivery of health services, remote patient monitoring, remote management of diseases, self-care systems, and patient tele-monitoring. These have proved that modern healthcare is pervasive and has become a scientific discipline [3]. We are now able to turn our traditional general practitioners' surgeries, clinical interventions, patient monitoring and public health protection into e-health services, delivered at any time, in any place with the involvement of empowered patients interested in self-management of their health. Despite the fact that security is a major issue in pervasive computing [7], people might be willing to compromise and give up a considerable amount of their privacy for the sake of medical treatment [8]. As the computing boundaries are extended and include physical spaces, people who are interacting with the devices are becoming aware of the amount of personal information, which is collected, exchanged and processed. In the health care domain, nevertheless, if provision of personal health information reassures people of their health and timely medical treatment when required, they might be more prepared to share personal healthcare information. The number of pervasive software applications built for the healthcare domain exceeds applications in any other domain.

In this paper we promote a layered Software Architectural (SA) style, which accommodates software components with computations based on OWL/SWRL enabled ontologies. Their purpose is to secure the delivery of remote and personalized healthcare services, based on reasoning. The SA has been illustrated through the scenario of self-care homes, where residents are remotely monitored and assistance guaranteed according to the interpretation and understanding of various situations they encountered in care homes. The novelty of the SA is in its core layers, which comprise specific software components with taxonomies, OWL ontologies and reasoning, for the purpose of (a) defining and describing a particular situation in care homes and (b) reasoning upon the most suitable service for that situation. However, the SA also specifies the exact set of software artifacts, spread across three layers, which have to be developed in order to achieve (a) and (b). These core layers fit very well within the known MVC pattern in software engineering, and thus software applications generated from the proposed SA are viable solutions for Web based applications. They can be developed in Integrated Development Environments (IDEs) and run on Cloud, iClouds, Android or similar operating

978-1-4799-2504-9/14 $31.00 © 2014 IEEE
DOI 10.1109/HICSS.2014.365

environments. The SA components have been deployed using NetBeans IDE, with Java Enterprise technologies for its front end. Accessibility to the OWL/SWRL enabled computations has been secured through the OWL-API plug-ins. Therefore the SA allows access to various types of persistence which may include relational databases and OWL concepts at the same time. It is important to note that SA components which store OWL/SWRL enabled computations are purely software engineering mechanisms for reasoning in remote delivery of healthcare and should not be confused with formal ontologies in Healthcare which create knowledge bases.

The paper is organized as follows. In the background section we describe what Self-care homes are and emphasize the pervasiveness of such environments and expectations we may have from them. The Scenario section sets the scene in order to introduce a particular situation in a care home, which would require the delivery of an appropriate service. In section 4 we introduce the proposed SA by describing its layers and computations within its core layers. We separately illustrate software components which house taxonomical structures and ontologies with their extensions and reasoning performed upon OWL concepts. We overview related work in section 5 and conclude in section 6.

2. The Background

Self **C**are **H**ome, SeCH, is a physical environment, in which residents who need constant care receive personalized services appropriate to their situation. These services can be for example r*ecommending* residents to take due medication or to stop a current social/physical activity; i*nforming* residents of any changes to their daily routine due to change of circumstances; a*ctivating a device* within SeCH automatically, such as switching a heater on in a resident's room; or i*ssuing an alarm* for the medical staff on duty because urgent medical attention is needed. To be able to provide these services, SeCH is equipped with sensors which detect the whereabouts of its residents and monitor their activities. In addition sensorized garments worn by residents may monitor their physiological changes. Services that are delivered as a result of the collation of information from the environment in SeCH are personalized. This means that sensors are not the only source of information in SeCH. Residents, when admitted to SeCH, state their preferences in terms of how they would like to make use of facilities, and how they would like to receive SeCH services. This means that software applications which support SeCH are less intrusive and more personalized. Sensor devices, objects embedded with computational capabilities, actuators that can activate or deactivate a device in SeCH and a communicator that residents use to interact with software applications are connected together through a wireless network.

3. The Scenario

Margaret, John, Peter and Paul are residents of SeCH. Their morning routine starts with having a shower, followed by breakfast, and taking morning medicine. Then, they have free time to take part in a physical and social activity suitable for their health condition and preferences. Every day a balance exercise class for senior residents takes place in the 'Function' room. Attendance is compulsory for geriatric residents, but other residents attend if they wish. Margaret usually takes part in the 'walking-for-all' activity in the adjacent park. This morning, however, she did not feel well and decided to go to her bedroom and read the daily paper. Margaret like all other residents of SeCH is being monitored. And the sensorized garment Margaret is wearing shows that she is feverish. When she was admitted to SeCH, Margaret indicated that she would prefer to have her allocated heater in her room turned on, if it is off, when she feels cold. Sensors in SeCH are to inform whether an allocated room is cold, normal or hot, considering the body temperature of the resident to whom the room is allocated, provided he/she is currently inside the room. These sensor devices indicate that the room is cold for Margaret.

Recommending, informing, activating a device, or issuing an alarm are actions which may be taken in SeCH in various situations. Considering that Margaret is feverish, and her rom is too cold for her, she would expect the heater in her room to be automatically turned on. Therefore, the expected service in this situation is "Activating a device". The software application which supports SeCH is therefore expected to trigger an actuator that turns on the heater in Margaret's room.

4. The ASeCS Architecture

Assistive **Se**lf **C**are **S**oftware (ASeCS) architecture is illustrated in Figure 1. It is component based and layered: each layer has its own purpose and role in the overall ASeCS architectural style.

The Context Management Layer (CML) and *Application Layer (AL)* have specific roles compared to other ASeCS layers. The ASeCS architecture

hosts software applications that primarily support SeCH and trigger the delivery of its services. Therefore a set of various software applications and their interfaces are stored within the AL. They communicate with software components stored in the lower ASeCS layers and interpret and manipulate any type of input and user interaction we may have in various situations in SeCH.

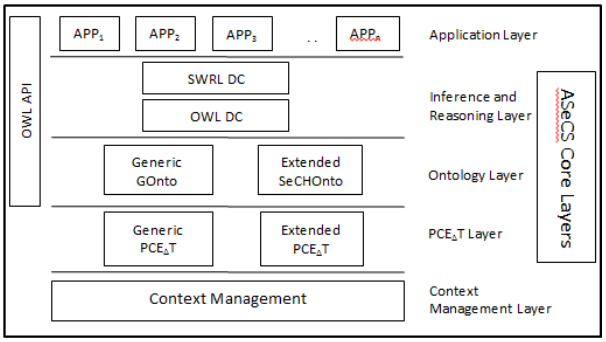

Figure 1: ASeCS software architecture

However, the CML has a completely different role. The availability of information about a particular situation in SeCH or "situational information" is essential to provide timely and appropriate service to the user. Acquiring "contextual data" from the environment is an indispensable part of any modern pervasive environment [9][10]. Sensitized garments, tagged heatesr, persistent data repositories, and software programs which integrate various devices into a pervasive environment, are examples of cyber-physical devices in SeCH which provide some form of contextual data to the ASeCS architecture. For a particular situation in SeCH, information on who the user is (Margaret), where she is, whether the room she is in is cold (given her body temperature), whether the heater in her room on or off are some of the "situational information" examples which define the situation in SeCH and deliver a service to fulfill Margaret's expectation. Consequently, such contextual data have to be managed, i.e. captured and interpreted [11][12][13][14]. The CML stores, represents and manages data received from the cyber-physical objects and prepares the "situational information" for the ASeCS upper layers. This is in line with many other similar solutions which require interpretation of the meaning of the collected contextual data, which has been exercised in context aware software applications for more than a decade [15][16][17]. For example, the detection of whether Margaret is "feverish" or not is the responsibility of the CML, and should be given to the upper ASeCS layers as a part of particular "situational information". Consequently, the CML makes sure that sensed data is qualified with semantics for further computation by ASeCS.

Computationally significant semantics provided by the CML is managed in the ASeCS core layers. They are taxonomical structures, denoted as $PCE_{\Delta}T$, OWL Ontology, and Inference/Reasoning mechanisms. It is important to note that the ASeCS core layers are essential for exploiting the situational information generated by the CML and delivering services through the applications App_n.

The *$PCE_{\Delta}T$ layer* stores taxonomies of a particular situation in SeCH. They may be very generic, i.e. they may contain basic taxonomical elements which could be used for describing any situation in SeCH. However, possible extensions of the *$PCE_{\Delta}T$* may be needed for two important reasons: (i) the situational information generated by the CML may require the creation of more situation-specific taxonomical elements in the *$PCE_{\Delta}T$* in order to secure the delivery of services in SeCH and (ii) generic taxonomical structure might not be sufficient to accommodate the specificity of the semantics in this particular domain (Healthcare).

All real world abstractions that participate in a situation in SeCH are accommodated in the taxonomical structure. In other words, the $PCE_{\Delta}T$ layer is responsible for arranging and organizing all detected taxonomical elements participating in the situation. However, the *$PCE_{\Delta}T$* might not have all abstractions ready for a particular situations and therefore extensions of the existing elements might be needed. For example, any real person in the SeCH without any specificity could be presented as an instance of **Psn** (Person). However, Margaret is a specific "Person" in SeCH. She is a **Resident** and thus the *$PCE_{\Delta}T$* should be extended: element **Psn** is extended into Resident to accommodate information relevant to Margaret. This is shown in Figure 2. "name, and "ender" are characteristics of Resident.

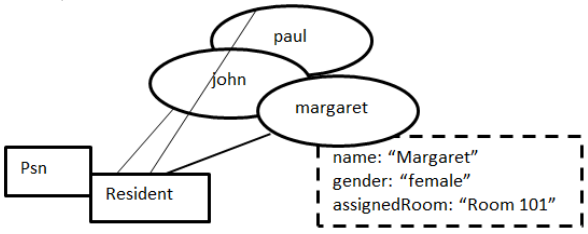

Figure2: Part of PCE_{Δ} T showing "margaret" in "Resident"

The *Ontology Layer* has a similar role to the *$PCE_{\Delta}T$ Layer*. The only difference is that the taxonomy of

the "situational Information" from the *PCE$_\Delta$T Layer* is transferred into OWL classes and properties. Similarly to the PCE$_\Delta$T Layer, the Ontology Layer hosts a generic OWL ontology GOnto (applicable to any situation in SeCH), which can be extended by adding the specificity of a particular situation in SeCH. Therefore the extended ontology is called SeCHOnto. GOnto represents a minimum of OWL concepts applicable to all situations in SeCH and SeCHnto represents the exact set of OWL concepts applicable to a particular situation.

The *inference and Reasoning Layer* provides an essential functionality for delivering services in SeCH. OWL ontologies are based on Description Logic and therefore inference on the concepts within GOnto or SeCHOnto is feasible using DL reasoning mechanism of OWL. However, when there is a need for reasoning about a complex semantic involving several concepts, OWL falls short and SWRL rules, which are also based on DL, have to be used. This is why the Ontology Layer is complemented with the Inference/Reasoning Layer to cover for the reasoning aspect of the ASeCS architecture.

Finally, the applications from the *Application Layer* are able to communicate with Ontology and Inference/Reasoning layers, through OWL-API. It ensures that software applications derived from ASeCS "know" SeCH inhabitants' precise location, their current activities, and present physiological vital sign measurements. They can "react" in order to assist residents in their everyday lives. In other words, "reacting" means delivery of personalized service(s) to SeCH residents.

4.1 Computations Defined in ASeCS

Computation in ASeCS that secures the delivery of a situation-specific service(s) in a particular situation is achieved by

1) creating a situation-specific taxonomical structure PCE$_\Delta$T for the situation and its counterpart in OWL ontology
2) reasoning upon the OWL elements in order to deliver a situation-specific service.

Consequently, the computations in ASeCS architecture have to secure the existence of a *generic taxonomical structure PCE$_\Delta$T*, which can fit any situation found in SeCH. However, the semantic of SeCH is always domain-specific and when dealing with a situation in SeCH we have to have domain and situation-specific taxonomical elements, as noted in 1). This implies that the computation should be able to create the exact PCE$_\Delta$T for each detected situation, i.e. to create a *situation-specific taxonomical structure*, which might have been extended from the generic PCE$_\Delta$T. The delivery of a situation-specific service must include computation that manipulates the semantics of the situation in SeCH through reasoning upon OWL elements of SeCHnto. Obviously the result of such reasoning is always a delivery of a situation-specific service(s).

It is important to note that the creation of *situation-specific taxonomical structure* is a powerful mechanism for delivering a correct service(s).

The reader should note that ASeCS architectural components cannot fully specify the exact computation of the delivered service(s), because services are domain and situation-specific. Figure 3 shows that for the given domain (this is Healthcare and we call it PCE$_\Delta$) a PCE$_\Delta$T (which is SeCH specific) is abstracted. We reason upon its taxonomical elements in order to deliver an expected service; i.e. triggering an actuator within SeCH that turns on the heater in Margaret's room.

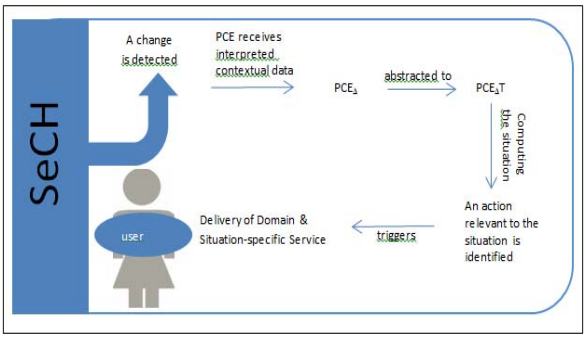

Figure 3: Computation of a situation-specific service in SeCH

4.2 Taxonomical Structure for SeCH

There are five essential taxonomical elements for SeCH.

OBJECTS - Given that SeCH is cyber-physical, it is naturally occupied with physical and tangible objects that do not necessarily bear any resemblance to a device. These objects could be tagged and equipped with appropriate microchips and sensor pads to act like a device, lending themselves to a more diverse SeCH. There are also intangible objects such as software programs which integrate various devices within SeCH or software applications which generate and manipulate data created within the SeCH. Each of these cyber and physical objects that are seamlessly interconnected through a wireless network, and allow pervasiveness of computing and communication of data at anytime and anywhere, have a purpose and role to play in

various situations. An abstraction of cyber and physical objects in SeCH is named **Ojt** for Object.

PERSON - In each situation in SeCH we know exactly which cyber physical objects are interconnected with the user, and what the user expects from the SeCH at that moment. User expectations are very often associated with services which should be delivered in a particular situation. However, there is always a user who is in charge of the SeCH. They can have distinguishing roles, and therefore should belong to different taxonomical element. Whatever the role of the user is, he or she is a person and thus we have **Psn** for Person.

FIELD - The third type of an abstraction is for all domain-specific information. Every service offered in SeCH is specific to the SeCH domain (Healthcare). We named such an abstraction **Fld** for Field.

These three basic elements of the $PCE_{\Delta}T$ might not be sufficient for describing SeCH. The elements that contribute towards the creation of a specific situation in SeCH often come from other abstractions such as PREFERENCES and LOCATION. Preferences **Pfc** of the user, and location **Lcn** of persons and objects are extremely important for delivering services. Their inclusion means that the software application which supports the user in SeCH is less intrusive and more personalized.

We have relationships between taxonomical elements in $PCE_{\Delta}T$. They are shown in Figure 4 and apply to SeCH in general. They are all self-explanatory. However, we may have a relation between "heater152", (Heater) and "Margaret" (Resident) which hasPreference relationship. It must exist between **Psn** and **Pfc**. However user may have preferences regarding his/her personal requirements, preferences related to objects and locations. This will result in defining a subset of Pfc namely **Ojt-specific-Pfc**, **Psn-specific-Pfc** and **Lcn-specific-Pfc**.

4.3. Extending the Taxonomical Structure

The *$PCE_{\Delta}T$* has to be extended to cater for any necessary abstractions contributing to the situation in SeCH. Possible extensions of the taxonomical elements from Figure 4 are shown in Figure 5: they are specific for SeCH Scenario given in Section 3.

The extension of Psn includes Resident, extension of Ojt includes Heater. The extension of the Lcn is "double": we need to know the exact Physical Location of a person and its private location (room allocated to a SeCH resident).

Fld may have any number of enumerated sub elements for a variety of domains such as health, education, manufacturing, business etc. In SeCH, the domain is health, public health, e-health and similar therefore Fld has extension Health, which further extends towards Care Homes and General Health in the SeCH situation described in the Scenario.

Taxonomical elements in Figure 5 colored with red are extensions of the generic *$PCE_{\Delta}T$*.

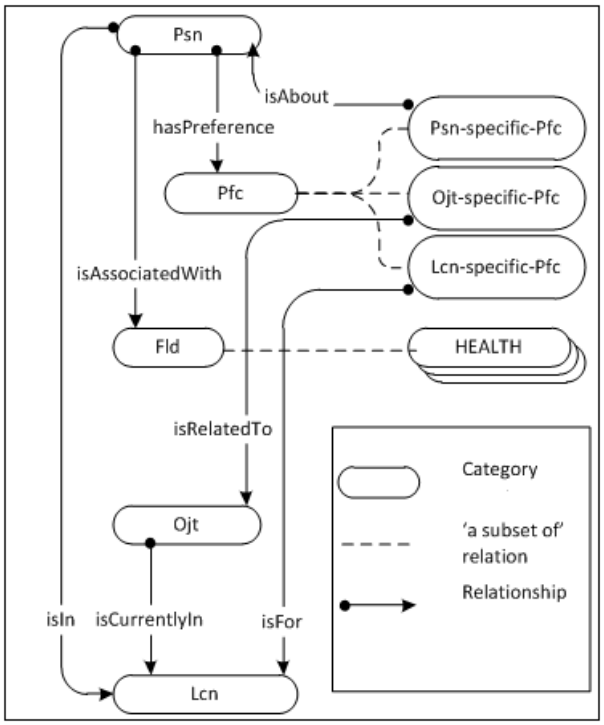

Figure 4: Summarization of generic $PCE_{\Delta}T$

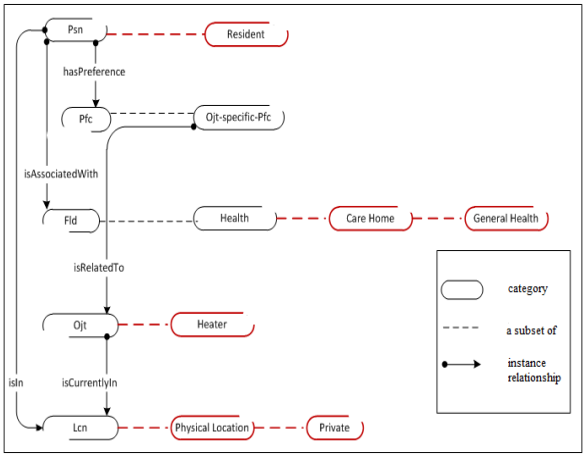

Figure 5: Extension of generic $PCE_{\Delta}T$

4.4 Creating GOnto

OWL ontologies have four concepts: individual, class, object and data type property. The variety of

features that OWL supports for each one of them is not the concern of this paper. What is important is to establish a mapping between each element of the $PCE_{\Delta}T$ and an OWL ontological model.

The counterpart of "element" in $PCE_{\Delta}T$ is "class" in OWL. Although classes can have different relationships with each other in an OWL ontology, here we are only interested in (a) the is-a (or subsumption) relationship, and (b) in relationships which are defined through object properties.

Every element of $PCE_{\Delta}T$ has some characteristics (see Figure 2). They cannot be shown graphically in Figures 4 and 5, because of space restrictions. In OWL they are data type properties. The domain of a data type property is their class. Their range value are different types, but we restrict them to "string" type.

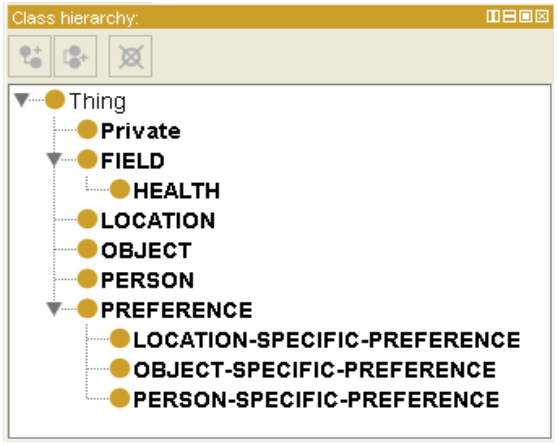

Figure 6: The generic GOnto OWL ontology

Object properties are also relationships defined by their domain and range, which by definition are both OWL classes. When individuals are asserted in OWL classes, if there are any relationships between them, object properties will be asserted.

Following the above, the generic $PCE_{\Delta}T$ shown in Figure 4 becomes an ontological hierarchy shown in Figure 6. We name this generic ontology GOnto. The name of classes in GOnto, unlike in $PCE_{\Delta}T$, have been deliberately chosen from English words to ease the writing, and interpretation of rules which are based on the ontological concepts and govern the delivery of services in SeCH. The object and data type property names remain as in the $PCE_{\Delta}T$ because they are self-explanatory.

4.5. Extending SeCHOnto

The availability of concepts in GOnto does not mean that every one of them has to be used. The situation in SeCH determines which ones are needed. The SeCHOnto, given in Figure 7 will provide the semantics for the delivery of expected service in the situation in SeCH described in the Scenario.

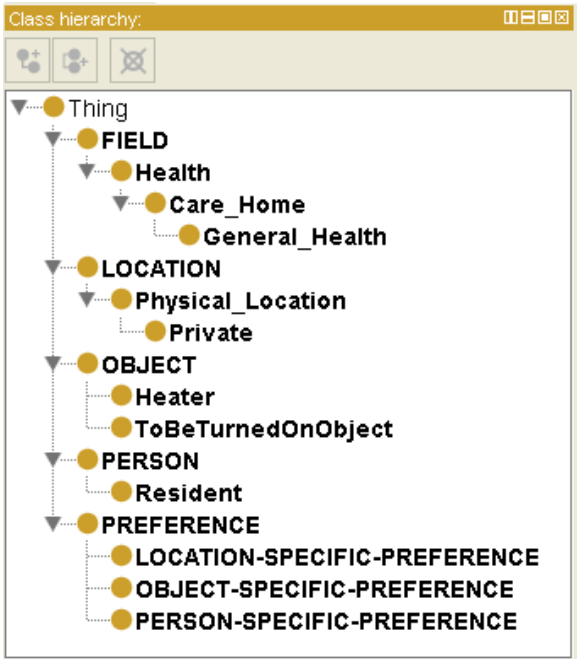

Figure 7: The SeCHOnto OWL ontology

Figures 6 and 7 do not show object properties between OWL classes. These are given in Table 1. All of them belong to GOnto, except "belongTo" which is situation-specific property for SeCHOnto.

Table 1: Generic and extended object properties

Object Property	Domain	Range
hasPreference	PERSON	PREFERENCE
isAssociatedWith	PERSON	FIELD
isRelatedTo	OBJECT-SPECIFIC-PREFERENCE	OBJECT
isCurrentlyIn	OBJECT	LOCATION
Isin	PERSON	LOCATION
belongsTo	OBJECT	RESIDENT

4.6. Inference with SWRL

Inference with SWRL has to be used for the purpose of decision making in order to deliver required services: to turn the heater in Margaret's room on. Once all the necessary OWL concepts have been identified, and the GOnto extended to SeCHnto, a reasoner engine can run the SWRL rule in order to

reason upon the assertions in SeCHOnto. It will infer new "knowledge" about already existing OWL individuals od SeCHOnto. We used the built-in Pellet reasoner [18] in Protégé 4.0 [19] ontology editor to run the SWRL rule. We show the final result of running the SWRL rule in Figure 8.

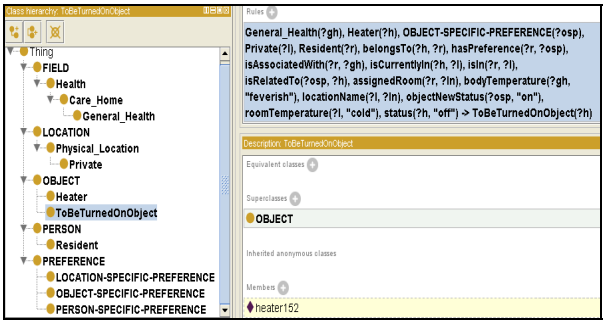

Figure 8: The result of SWRL rule

The SWRL rule that reasons upon the semantics in SeCHnto is shown in Table 2. The premises of the rule represent the semantics of a situation in SeC described in the Scenario. It consists of a number of atoms. Some of them are individuals: Resident(?r) represents a typical resident "r" and Location(?l) represents a typical location "l". Other atoms represent the binary relationships between two individuals. For example, isIn(?r,?l), represents relationship "isIn" between two typical individuals "r" and "l" which are of course defined before being used in this relationship. The conclusion of the rule is the result of the reasoning: class ToBeTurnedOnObject(?h) indicates that individual "h", which is defined in the premise as a Heater, will also be an individual of the class ToBeTurnedOnObject.

Table 2: SWRL rule for the example scenario

General_Health(?gh),Heater(?h),Location(?l),Object-Specific- Preference (?osp), Resident(?r), belongsTo(?h,?r) , hasPreference(?r,?osp), isAssociatedWith(?r,?gh), isCurrentlyIn(?h,?l), isIn(?r,?l),isRelatedTo(?osp,?h), assignedRoom(?r,?ln), bodyTemperature(?gh,"feverish"), locationName(?l,?ln), objectNewStatus(?osp,"on"), roomTemperature(?l,"cold"), status(?h,"off")-> ToBeTurnedOnObject(?h)

5. Related Works

There are no available publications on SA which use specific core layering for accommodating OWL/SWRL enabled computations.

However, OWL ontologies are increasingly being used in the health domain, mostly for vocabulary specification or for serving context aware applications. Examples for the former are Gene Ontology Consortium, with the role of producing a dynamic, controlled vocabulary for genes [20], a big biomedical vocabulary like a Thesaurus for cancer research [21], large scale clinical terms SNOMED [22], or examples which serve the needs of a particular community such as phenotype ontologies [23]. Ontologies which are used for purposes other than vocabulary, and in environments similar to SeCH are [24][25][26][27][28][29][30][31][32][33]. Most of them have already been formalized in healthcare, but they can still be retrieved and used in software applications generated from the ASeCS architectures, if necessary. It allows the retrieval of the ontological elements through OWL-API.

It is important to note that there are also numerous solutions which use ontologies for managing data which can be stored in the ASeCS Context layer. Chen and Finin [34] consider ontologies as 'key requirements' for building context-aware software applications and in Chen et al [35] they developed a shared ontology SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications) for supporting pervasive software applications. They believe that their generic ontological model in OWL, can be a step towards the standardization of a shared ontology to be reused by ontology-driven application developers. Wang et al. [25] use a set of ontologies to describe and represent contextual information within their SOCAM architecture. In their ontology based U-HealthCare, Ko et al. [26] have defined three context ontologies for Person, Device and Environment, and their model does not include the element of time. Their ontologies are semantically divided into general context ontologies and domain context ontology, similar to [25].

Paganelli and Giuli [27] provide a more detailed ontological model than [26] for their 'Kamer' project. They have provided four ontologies to represent patients, other people patient encounters, the physical environment, and alarm management ontology. In the 'Patient Personal Domain Ontology' they include patient physiological information. They use OWL and some first order logic rules to reason upon the "context".

It is important to note that all these solutions do not house OWL/SWRL combined code within their architectures for the purpose of deploying them as a computational solution for delivering services in Healthcare. Their OWL repositories address a

complete environment and they are not "situation" specific.

6. Evaluation and Conclusions

The proposed ASeCS architecture's specificity is in the extensive use of Semantic Web Technologies (SWTs) and OWL/SWRL enabled computations in particular, which dictate the nature of its software components and architectural layering. The SA is technology specific because it has to guarantee the deployment of its components. It is also designed with "the delivery of healthcare services" in mind, because its core layers secure the interpretation of the semantics stored in environments built for remote patient monitoring. We would like to justify our architectural design in the next few paragraphs.

If we wish to create a new era of software engineering solutions based on the semantic and understanding of our computational environments in modern healthcare environment, the use of SWTs is the way forward. The SWT stack [37] has been created with the semantic "Web" in mind, but the same philosophy, ideas and languages in particular can be re-used outside the semantic Web domain. If we can transfer the interpretation of and reasoning upon the content of the Web to any computational environment, then we can achieve an almost identical result as on the Web. In our particular domain of *creating situations in SeCH and reasoning upon it in order to deliver services*, we need the same mechanism: describe the domain ("situation" within a SeCH) using SWT languages, and reason upon it using SWRL in order to create a computational result ("deliver a service").

The choice of OWL sublanguages within the SWT stack is impressive. Of the three sublanguages: OWL Lite, OWL DL and OWL Full, we have used OWL DL. Considering that the purpose of the computation in SeCH is to reason upon taxonomical elements of $PCE_{\Delta}T$ to deliver a service to the user of SeCH, the variant of OWL to be chosen should support SWRL as the reasoning language used in the SWT stack. This requirement automatically dismissed OWL Full, which was not a suitable candidate because of its unrestricted expressivity. OWL Full allows restrictions to be defined at the "meta level" and therefore types, such as classes and individuals, are not separated from each other. Elements of $PCE_{\Delta}T$, on the contrary, are clearly separated from each other and therefore OWL Full could not be suitable for mapping in our case.

Furthermore, computations in environments such as SeCH should be computationally "cheap". This means that building any knowledge base and excessive persistence, which could underpin the delivery of services in SeCH is out of the question, i.e. knowledge bases are not essential in deploying ASeCS components. In the era of highly accessible mobile and wireless computing, we should assume that all our implementations should run on mobile devices. Therefore, we should have the possibility of hosting software applications generated from ASeSC in various Clouds.

The ASeCS architecture has been tested in various applications of the SeCH domain, which were developed with NetBeans. OWL-API was used for accessing OWL/SWRL enabled computations, where the Protégé editing tool and reasoners were involved. The front end of our applications have been programmed in JavaServer Pages and we used a selection of components from the AL, deployed with Servlet Technology in order to manage the applications [38]39][40]. The experiences are summarised below.

The use of SWTs through IDEs such as NetBeans, shows that we are able to extend the same mechanism of manipulating the semantics of the Web towards any other form of computations in software engineering, which does not have to be related to the Internet. However, we have to bear in mind that traditional software technologies, including Java technologies, cannot manage reasoning in SeCH without reference to SWT. Software applications needed by SeCH cannot rely only on procedural or object-oriented programming languages alone to address requirements of SeCH. Managing them through knowledge base systems and making them dependent on constantly growing persistence would not satisfy a fraction of expectations we have from environments such as SeCH.

Although in real life situations all SWRL rules are usually defined in advance and stored with ontologies such as GOnto in our case, we have also experienced how SWRL rule can be defined, created and executed through the Application Layer at run time once SeCHOnto has been created. Readers should notice that the result of the inference and reasoning of a situation in SeCH is just "for the moment", when the situation in SeCH occurs, and as soon as another change in SeCH is detected, the result of the reasoning related to the previous "moment" has to be deleted. This is because the inference/reasoning of a particular situation in SeCH might not be exactly correct contextual information or suitable for another situation. This means that each time a change is detected in SeCH, software applications generated from the ASeCS architecture must reload GOnto and disregard situation-specific SeCHOnto once the reasoning on the situation which delivered a service

is done. Nevertheless, accessing GOnto from outside of Protégé to extend it to SeCHOnto requires somewhat frustrating interaction with confirmation dialog boxes, because the current version of Protégé does not support its OWL file to be handled by applications automatically.

Although Protégé 4 is user-friendly and the most commonly-used open source ontology editor, it is still an incomplete and somewhat unstable tool. The obvious example is the "tab" provided in Protégé 4, for editing SWRL rules. There is a limit to the number of atoms one can employ in each rule. If the number exceeds the limit, a number of the "consequences" in the rule, would literally be omitted from the rule's syntax.

There is scope for improvements and future works. SeCH is an example of a very dynamic pervasive healthcare environment and we must allow its extension, removal and replacement of devices without any restriction. This requires devices to be self-maintaining in terms of the meaningful data they provide. The integration of devices in SeCH and their management is the area which would need attention in future. The ASeCS architecture does not necessarily acknowledge this problem because it focuses on the interpretation of contextual data when collecting the semantics of a situation in SeCH. It remains to be seen if ASeCS would change in order to accommodate the dynamics of cyber-physical objects in environments like SeCH.

We should look at implementations of the ASeCS architectures in environments with Android operating systems. The lightness of our computational solution defined in the ASeCS core layers is encouraging. It remains to be seen how we can maintain the MVC pattern, which is so prevalent in modern computing and accommodate the dynamic environments where Apps are developed and programmed.

7. References

[1] Arnrich, B., Mayora, O., Bardram, J. and Tröster, G. (2010) '*Pervasive healthcare: Paving the way for a pervasive, user-centered and preventive healthcare model*', *Methods of Information in Medicine*, 49:1, pp. 67-73.

[2] Coronato, A. and Pietro, G.D.E. (2010) '*Formal specification of wireless and pervasive healthcare applications*', *ACM Transaction in Embedded Computing Systems,* 10:1, Article 12.

[3] Bardram, J.E. and Christensen, H.B. (2007) '*Pervasive Computing Support for Hospitals: An overview of the Activity-Based Computing Project*', *IEEE Pervasive Computing*, 6:1, pp. 44-51.

[4] Romero L.M.R., Tosina L. J. R., Valderrama M.A.E., Arbizu J.C. and Martíne I.R. (2011) '*A Comprehensive View of the Technologies Involved in Pervasive Care*', *Communications* in *Medical* and *Care Compunetics*, pp. 3-19.

[6] Zhang Y., Jia Z. and Chen Y. (2011) '*A thought on the goals & realization of pervasive healthcare*', Scientific Research & Essays, Vol. 6 (13), pp. 2752-2756.

[7] Campbell, R, Al-Muhtadi, J, Naldurg, P, Sampemane, G, Mickunas, M (2002) '*Towards security and privacy for pervasive computing*', ISSS, Tokyo, Japan, pp. 1-15.

[8] Bohn, J., Gartner, F. and Vogt, H. (2004) '*Dependability issues of Pervasive Computing in a healthcare Environment*'.

[9] Schmidt, A. (2000) *'Implicit Human-Computer Interaction through Context*', Personal Technologies, 4 (2 & 3), pp. 191-199.

[10] Henricksen, K. and Indulska J. (2006) '*Developing Context-Aware Pervasive Computing Applications: Models and Approach*', *Pervasive and Mobile Computing*, 2, Issue 1, pp. 37-64.

[11] Dey, A. K. (1998) '*Context-Aware Computing: The CyberDesk Project*', *AAAI 1998 Spring Symposium on Intelligent Environments*, Technical Report SS-98-02, pp. 51-54.

[12] Dey, A. K. (2001) '*Understanding and using context*', *Personal and Ubiquitous Computing*, 5, No.1, pp. 4-7.

[13] Strang, T. and Linnhoff-Popien, C. (2004) *'context modelling survey*', In 1st Int. Workshop on Advanced Context Modelling, Reasoning and Management. *Comm*, 6, No. 8, pp. 10–15.

[14] Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A, Riboni, D. (2010) '*Survey of Context Modelling and Reasoning Techniques*', *Pervasive and Mobile Computing*, 6, Issue 2, pp. 161-180.

[15] Gu, T., Wang, X., Pung, H. and Zhang, D. (2004) *'An Ontology-based Context Model in Intelligent Environments*', In Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, California, USA.

[16] Ellenberg, J., Karstaedt, B., Voskuhl, S., Luck, K.V, and Wendholt, B. (2011) '*An environment for context-aware applications in smart homes*', in to appear in: International Conference on Indoor Positioning and Indoor Navigation (IPIN), Guimaraes, Portugal.

[17] Sang, P., So, W., Jong, L., Sung, K. (2003) '*Smart home – digitally engineered domestic life*', Personal and Ubiquitous Computing. pp. 189-196.

[18] Pellet: OWL 2 Reasoner for Java,*2004* available at http://clarkparsia.com/pellet/ (accessed October 2012)

[19] Protégé Documentation, Protégé-owl api, 2009, available at *http://protege.stanford.edu/plugins/owl/api/* (accessed October 2009).

[20] GO (2000), Gene Ontology, The Gene *Ontology Consortium: tool for the unification of biology*. Nature Genet, 25, pp. 25-29.

[21] Hartela, F., W., Coronadoa, S., Dionneb, R., Fragosoa, G., Golbeckc, J. (2005) '*Modelling a description logic vocabulary for cancer research*', *Journal of Biomedical Informatics*, 38, pp. 114-129.

[22] Spackman, K. (2000) *'SNOMED RT and SNOMED CT*', Promise of an international clinical ontology, M. D. Computing 17.

[23] Mungall, C.J., Gkoutos, G. V., Smith, C.L., Haendel, M. A., Lewis, S. E., Ashburner, M. (2010) 'Integrating phenotype ontologies across multiple species', *Genome Biol*, 8, 11(1).

[24] Chen, H., Finin, T., Joshi, A. (2003) '*Using OWL in a Pervasive Computing Broker*'.

[25] Wang, X.H., Zhang, D.Q., Gu, T., Pung, H.K. (2004) '*Ontology based context modelling and reasoning using OWL*', Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on, pp. 18-22.

[26] Ko, E. J., Lee, H. J. and Lee, J. W. (2007) '*Ontology-Based Context Modeling and Reasoning for U-HealthCare*', *IEICE - Trans. Inf. Syst.* E90-D, pp. 1262-1270.

[27] Paganelli, F. and Giuli, D. (2007) '*An Ontology-Based Context Model for Home Health Monitoring and Alerting in Chronic Patient Care Networks*', Proceedings of the 21st international Conference on Advanced information Networking and Applications Workshops, AINAW. IEEE Computer Society, Washington DC, pp. 838-845.

[28] Bardram, J.E. and Christensen, H.B. (2007) '*Pervasive Computing Support for Hospitals: An overview of the Activity-Based Computing Project*', *IEEE Pervasive Computing*, 6:1, pp. 44-51.

[29] Niyato, D., Hossain, E., Camorlinga, S. (2009) '*Remote Patient Monitoring Service using Heterogeneous Wireless Access Networks: Architecture and Optimization*', *IEEE Journal on Selected Areas in Communications,* 27:4, pp. 412-423.

[30] Arnrich, B., Mayora, O., Bardram, J. and Tröster, G. (2010) '*Pervasive healthcare: Paving the way for a pervasive, user-centered and preventive healthcare model*', *Methods of Information in Medicine*, 49:1, pp. 67-73.

[31] Polze, A., Tröger, P., Hentsche, U., Heinze, T. (2010)*13th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops'*, pp. 204-210.

[32] Coronato, A. and Pietro, G.D.E. (2010) '*Formal specification of wireless and pervasive healthcare applications*', *ACM Tansaction in Embedded Computing Systems,* 10:1, Article 12.

[33] Romero L.M.R., Tosina L. J. R., Valderrama M.A.E., Arbizu J.C. and Martíne I.R. (2011) '*A Comprehensive View of the Technologies Involved in Pervasive Care*', *Communications* in *Medical* and *Care Compunetics*, pp. 3-19.

[34] Zhang Y., Jia Z. and Chen Y. (2011) '*A thought on the goals & realization of pervasive healthcare*', Scientific Research & Essays, Vol. 6 (13), pp. 2752-2756.

[35] Chen, H. and Finin, T. (2003) '*An Ontology for Context-aware Pervasive Computing Environments*'.

[36] Chen, H., Perich, F., Finin, T. W. and Joshi, A. (2004b) '*SOUPA: Standard ontology for ubiquitous and pervasive applications*', Proceedings of the First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04), IEEE Computer Society, pp. 258-267.

[37] W3C (2004) '*OWL Web Ontology Language Overview*', Available at: http://www.w3.org/TR/owl-features/ (Accessed: 8 March 2013).

[38] Shojanoori, R., Juric, R., and Tourani, B. (2010) '*Experiences of building assisted self-care systems within smart home environment'*. In: Proceedings of the 15th International Conference on System Design and Process Science, 06 – 11, Dallas, USA.

[39] Shojanoori, R., Juric, R. and Lohi, M. (2012) *'Computationally Significant Semantics in Pervasive Healthcare'*, Transactions of the SDPS: Journal of Integrated Design and Process Science, 16 (1), 43-62.

[40] Shojanoori, R. and Juric, R. (2013) *'Semantic Remote Patient Monitoring System'*, Telemedicine and e-Health, 19 (2), 1-8.