Computer and Information Science; Vol. 11, No. 2; 2018
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

A Synergy of Semantic and Context Awareness for Service Composition in Ubiquitous Environment

Tarik Fissaa[1], Hatim guermah[1], Mahmoud El hamlaoui[1], Hatim Hafiddi[2] & Mahmoud Nassar[1]

[1] IMS Team, ADMIR Laboratory, ENSIAS, Rabat IT Center, Mohammed V University, Rabat, Morocco

[2] ISL Team, STRS Lab, INPT, Rabat, Morocco

Correspondence: Tarik Fissaa, IMS Team, ADMIR Laboratory, ENSIAS, Rabat IT Center, Mohammed V University, Rabat., Morocco. Tel: 212-672-943-348. E-mail: Fissaa.Tarik@gmail.com

Received: April 6, 2018 Accepted: April 17, 2018 Online Published: April 30, 2018

doi:10.5539/cis.v11n2p88 URL: https://dx.doi.org/10.5539/cis.v11n2p88

Abstract

Service composition in an important facet in service oriented architecture, it's about the idea of assembling atomic services to satisfy a demand rather than building new applications from 'scratch', From the user's perspective it's a complex task due to the increasing number of services in the web and their heterogeneity. This complexity is increasing in the internet of Things era wehere computing devices are everywhere. In this work we propose an approach for composition of context aware services in a semantic manner, Artificial Intelligence planning is used to automate the composition starting from a defined objectif containing user request and context parameters. Service are described by extending OWL-S with contextual conditions. The proposed architecture was evaluated through an e-health scenario where chronic patients can benefit from a remote and automated medical supervision and emergency handling.

Keywords: context awareness, service composition, artificial intelligence, semantic web services

1. Introduction

Of the actual computing era of internet of thing where multiple connected devices pervades the environment, this is the results of the development in many research areas such wireless networks, artificial intelligence, small devices the computing power of actual computers. All these advances lead to a new generation of services based on personalization recommendations and adaptability. Context aware services provide users with customized behaviors according to everyone's surrounding context

One of the core principles of Service-Oriented Architectures (SOA) is the idea of assembling services to form a chain by composing those multiple existing services to satisfy a user task, rather than building new complex applications from 'scratch'. Automation of this process is emerging as one of the most interesting and complex challenges facing SOA today due to the amount and increasing number of services (Rao, J., & Su, X, 2004).

On the other hand, to be context-aware composite services need to follow some requirements to resolve the challenges brought by the context-awareness paradigm, thus the composed service should be built in an automatic and dynamic way depending on the context of use.

To tackle those problems, a synergy approach between web semantic paradigm and self-adaptive systems is necessary (Sheshagiri, M et al, 2004). In one hand semantic web services through preconditions and effects are needed to automate the composition and context awareness to add dynamicity and adaptability to the service composition.

To highlight the fundamental challenges for the development of context-aware service composition, an E-health motivating scenario is presented as a walk-through example. The objectif is to give people with chronic disease a certain freedom and independence in their daily life by being monitored remotely by a context aware e-health system. A patient is continuously monitored to detect crises and problems. Thus if a problem is detected the patient, doctors and relatives will be informed in real time and a number of actions(services) will be called to deal with the situation in hand. The role of the system is to find ambulance, hospital, doctors according to several context parameters (location, activity…). The figure 1 explains some features of the system.

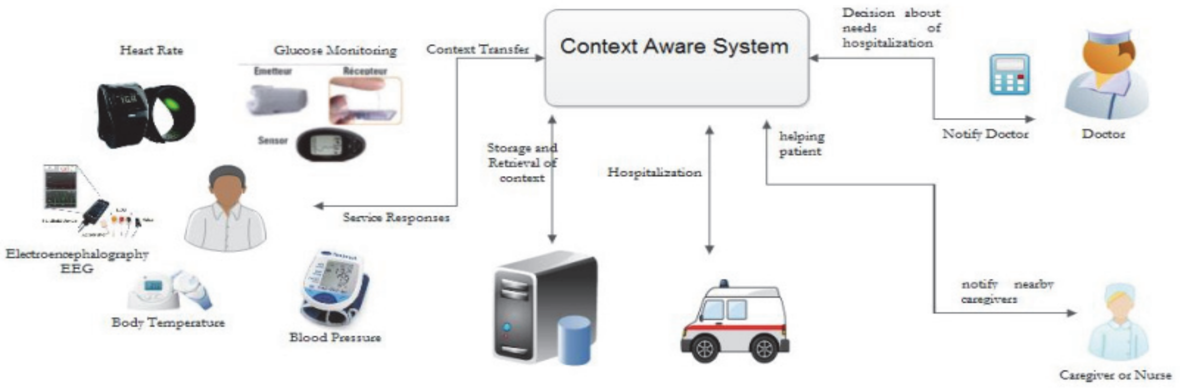


Figure 1. E-health scenario

There are several approaches for service composition techniques such as workflow and artificial intelligence (AI) planning (Peer J, 2005). Workflow approaches are mainly used with a well-defined, manually built process model, but not in a situation where there is a need to find the suitable atomic services in an automatic way without having in advance a predefined process of services. On the other hand, AI planning techniques allow making a plan that best fits to the users' preferences and restrictions without having a fixed process model by using semantically machine-interpretable information. Since ubiquitous computing environments are very dynamic and unpredictable, AI planning methods are more suitable and useful for creating dynamic service composition than workflow methods.

To tackle the problems listed above, this work proposes an architecture for developing composite context-aware services. Contributions can be summarized as follows:

- Context Modeling: context modeling is an important issue in ubiquitous systems to allow context awareness and adaptability. For this an ontology based context modeler was proposed
- Service Specifications: Actual service specifications do not allow taking context into considerations. Therefore, an extension to the OWL-S (Ontology Web Language for Services) is proposed, named 'CA-OWLS'. This extension allows the representing of contextual constraints.
- An architecture for developing composite context-aware services. The architecture consists of three layers: Goal Context Manager, Context aware service composition, Execution and Monitoring.
- Use Case: an e-health scenario was chosen to highlight the different steps of our application.

This paper envisages a novel application architecture to support adaptivity in the composition process by means of an ontology language and related tools for context-awareness in service design.

2. Method

This section describes how the composition process is conducted. First we describe the context ontology and the service description by extending OWL-S, then, we describe the composition architecture layers, and how each layer works.

2.1 Ontology Modeling

We choose to model context element by using an ontology containing different context parameters, the context ontology is composed of the following sub contexts:

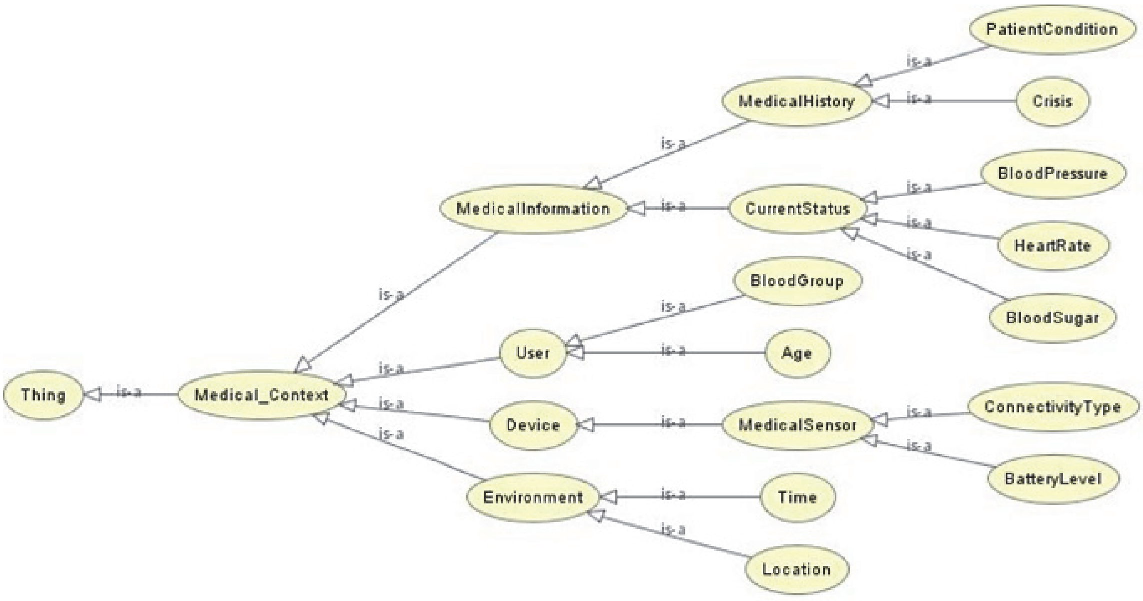


Figure 3. Excerpt of the OWL context ontology in Protégé editor

- Environnent: location, time.
- Medical: contains user’s health properties (antecedents, medical files…etc.).
- User: contains properties describing the user (vital signs, activity, user preferences).
- Device: contains parameters that describe the entity Device (e.g. medical devices, Car system, Smartphone, etc.).

2.2 Context Aware Service Description

OWL-S (Martin D et al, 2007) is an ontology for the description of semantic Web services expressed in the Web Ontology Language (OWL). OWL-S defines an upper ontology for semantically describing Web services along three main aspects: The Service Profile describes what the service does in terms of inputs, outputs, preconditions and effects (IOPEs). The Service Model describes how a service works in terms of a process model that may describe a complex behavior over underlying services, it distinguishes between atomic, simple and composite processes. The Service Grounding describes how the service can be accessed, usually by grounding to WSDL. But to take advantage of context-awareness it’s important to have an efficient mechanism to adapt services (composite or single ones) according to the context, which is not supported by the current OWLS. Thus an extension to OWL-S was proposed, based on context elements, to detect the necessary adaptations.

Is a service ontology for modeling semantic in web services, it’s based on OWL. OWL-S is composed of an upper ontology with three parts, the service profile is for discovery and contain IOPE. The service Model represents the process, the process can be atomic, simple or composite. The service grounding part is for accessing and invoking the service from its WSDL. In our work we added a mechanism to adapt OWL-S to context aware environment by adding context conditions and parameters which was not supported by the original OWL-S

Figure4 illustrates the proposed context extension (CA-OWLS). The CA-OWLS specifications are:

- The CA-OWLS has a ServiceModel, ServiceProfile and ServiceGrounding.
- The service model represent a process.
- The Process contains AtomicProcess, CompositeProcess or SimpleProcess.
- The ServiceProfile is related to a context Property.
- Each ContextProperty contains Contextual Attributes.
- Adaptation condition: The service may require external pre-condition to be satisfied to execute the process.
- Adaptation Effect: The execution of the service may result in certain external effects.

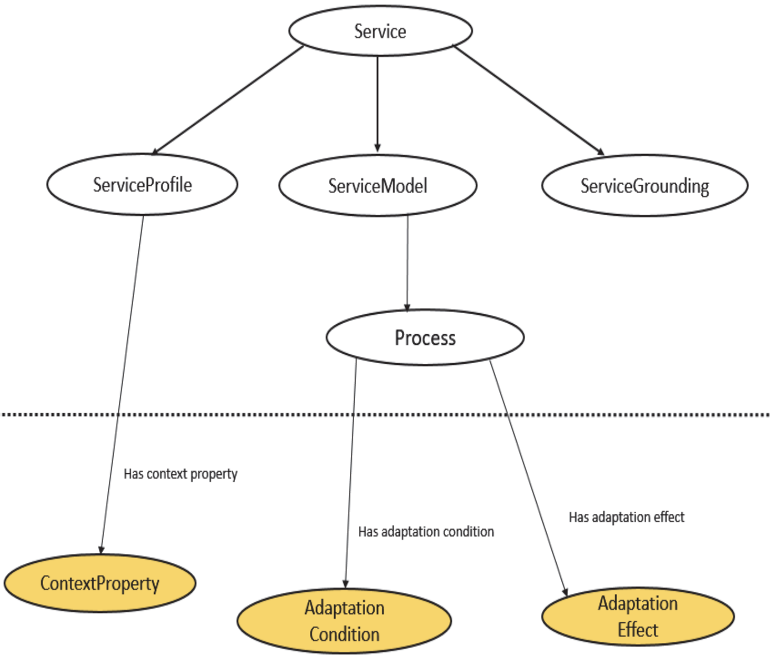


Figure 4. OWL-S Extension with context elements (CA-OWLS)

2.3 Architecture of the Composer

In this section we present the context aware service composition architecture. Several services in a repository are assembled to form a plan of services, those plans are adaptive to every user according to their context. If the context changes, the composite service changes in an automatic and dynamic manner. Several layers compose our architecture, which are:

- Context Goal Manager Layer.
- Context aware composition layer.
- Execution and Monitoring layer.

2.3.1 Context Goal Manager Layer

The contextual request management is the first component of the system. Figure 5 shows the structure of this layer. The user or an agent on his behalf send a contextual request. This request is personnlized by adding context element for every consumer. Then, the composition system respond to this contextual demand.

If the context changes during the composition, the control is passed back to the composition context goal manager layer, which will transform the user request into an alternative one.

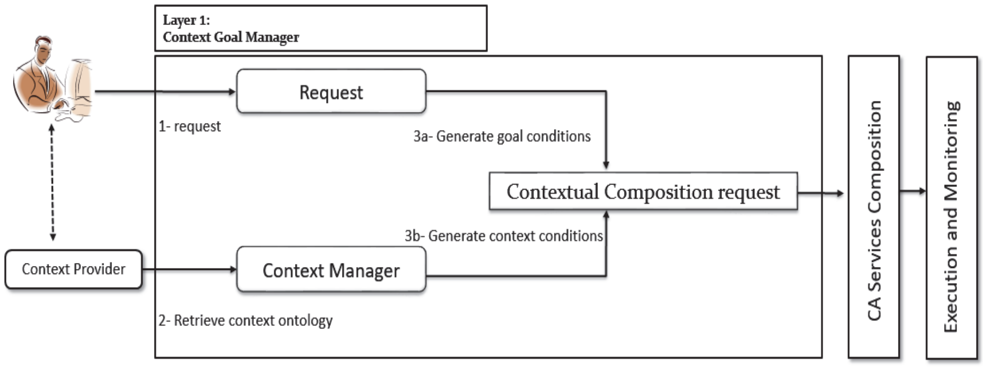


Figure 5. The context goal manager layer

2.3.2 Context Goal Ontology

The description of the goal that represents the user task or request is given in the form of an abstract CA-OWLS process, in order to take into account the contextual conditions in the initial state and goal state (Figure 6).

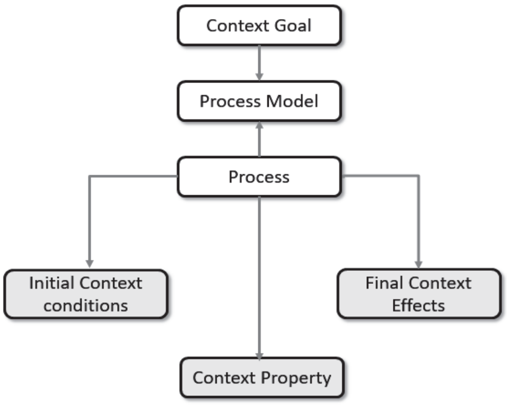


Figure 6. Context Goal Ontology

The main difference between the context goal descriptions and service descriptions is that in contrast to the atomic processes involved in the services processes those involved in user task processes are not bound to any concrete service, since services to be invoked are dynamically discovered. Thus the OWL-S Grounding corresponding to the request is generated at runtime from the Groundings of the composed context aware services.

The context manager collects information from different entities that affect context (e.g., sensors, users, devices, etc.). Then, it uses a context ontology to provide contextual information formatted in this ontology.

The main difference between the context goal descriptions and service descriptions is that in contrast to the atomic processes involved in the services processes those involved in user task processes are not bound to any concrete service, since services to be invoked are dynamically discovered. Thus the OWL-S Grounding corresponding to the request is generated at runtime from the Groundings of the composed context aware services.

The context manager collects information from different entities that affect context (e.g., sensors, users, devices, etc.). Then, it uses a context ontology to provide contextual information formatted in this ontology.

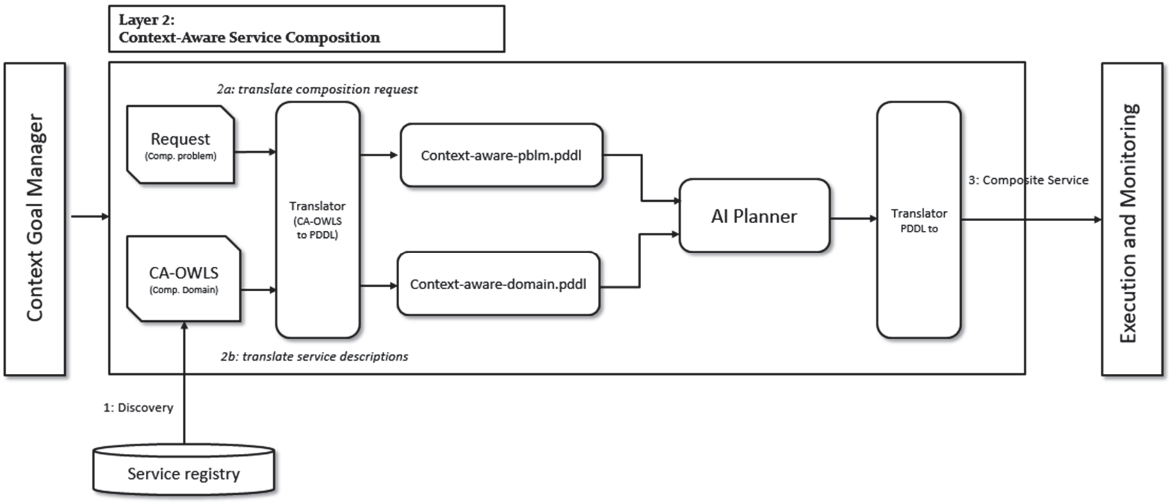


Figure 7. Context-Aware Service Composition

2.3.3 Context Aware Composition Layer

In this layer, after receiving a personnlized request, an AI planning module is called to perform a planning of a set of services in order to respond to the user demand. This layer is composed of the following modules as show in Fig 6:

- Translator (CA-OWLS2PDDL): Translate context aware services and context goal ontology to a domain and problem respectively in PDDL (McDermott, D., Ghallab, 1998).
- CA-Service Composition: an AI planning algorithms performs the planning phase.
- Translator (PDDL2CA-OWLS): Translate a plan of actions in PDDL into a CA-OWLS

a) CA-OWLS to PDDL:

The role of this component is to transform the set of CA-OWLS descriptions and the context goal ontology into a domain and problem planning respectively.

Every web services in a repository is called to create a corresponding PDDL action. The analogy between semantic web services and PDDL help to create the actions in order to use them by a planner.

Every other parameter (IO and context elements) are transformed into additional preconditions and postconditions. To this aim, we create a predicate ("*Has Knowledge"*) defined in the ontology to facilitate the inclusion of IO and context in the composition process. For example the context parameter location we create a predicate "*hasKnowledge(Location)"* as additional information (Figure 8a).

The initial state (respectively the goal state) of the context goal ontology is translated to equivalent initial and goal state in the planning problem (Figure 8b).

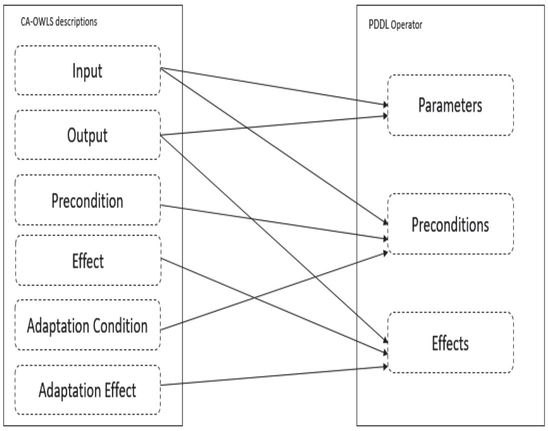


Figure 8a. Domain translation

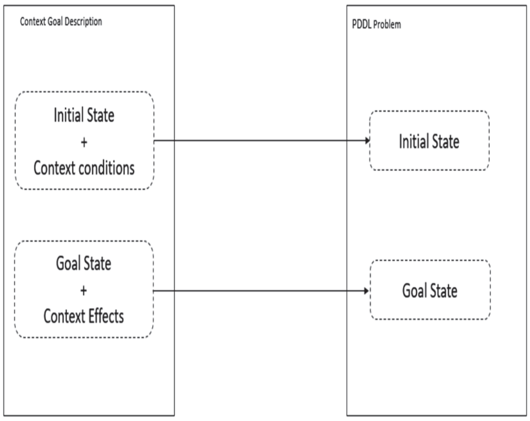


Figure 8b. Problem translation

b) Automatic service composition: AI Planning select the best actions in a certain domain and create plans to solve an objectif starting from an initial state. A classical AI planning for service composition problem can be formalized as a quintuple $< S; S_0; G; A; T>$, where:

- S is the set of all possible facts.
- $S_0 \subset S$ denotes the initial state.
- $G \subset S$ denotes the goal state.
- A is the set of services
- $T \subseteq SxAxS$ a transition function

A state is a set of preconditions or a set of postconditions (effects). They are represented as predicates in both the initial and objectif state.

For transiting between different states, an action is executed causing a change in the knowledge (predicate changing in preconditions and effects both for ordinary parameters and context parameters)

So we have two type of states ordinary ones and context ones. Ordinary state represent functional parameters, and also preconditions and effects of those parameters. The contextual states can be accessible by users, devices…and they are not essential for the system. Their role is to personalize services and enhance the user experience, it includes all context information according to Dey's context definition

c) Execution and Monitoring Layer:

The Execution Layer is the module responsible of calling and executing services, the Monitoring is bound to Execution Module to check every time for external events, such as context changes, disponibility of services. If context changes this lay sends information to the first layer to change the context request or to second layer if a service is not available.

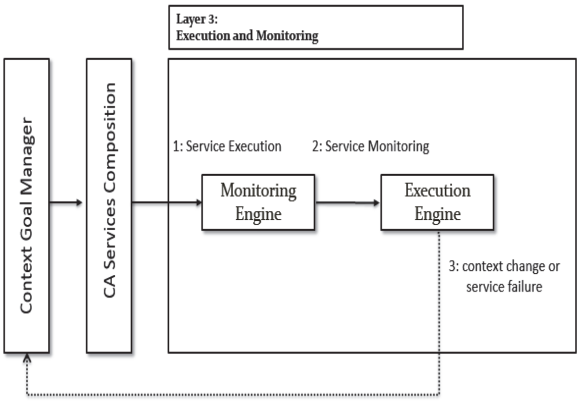


Figure 9. The Execution and Monitoring Layer

3. Results and Discussion

To illustrate how context aware compositions can be built, our approach was tested on a simplified version of the application scenario (Emergency Monitoring) introduced at the beginning of this paper.

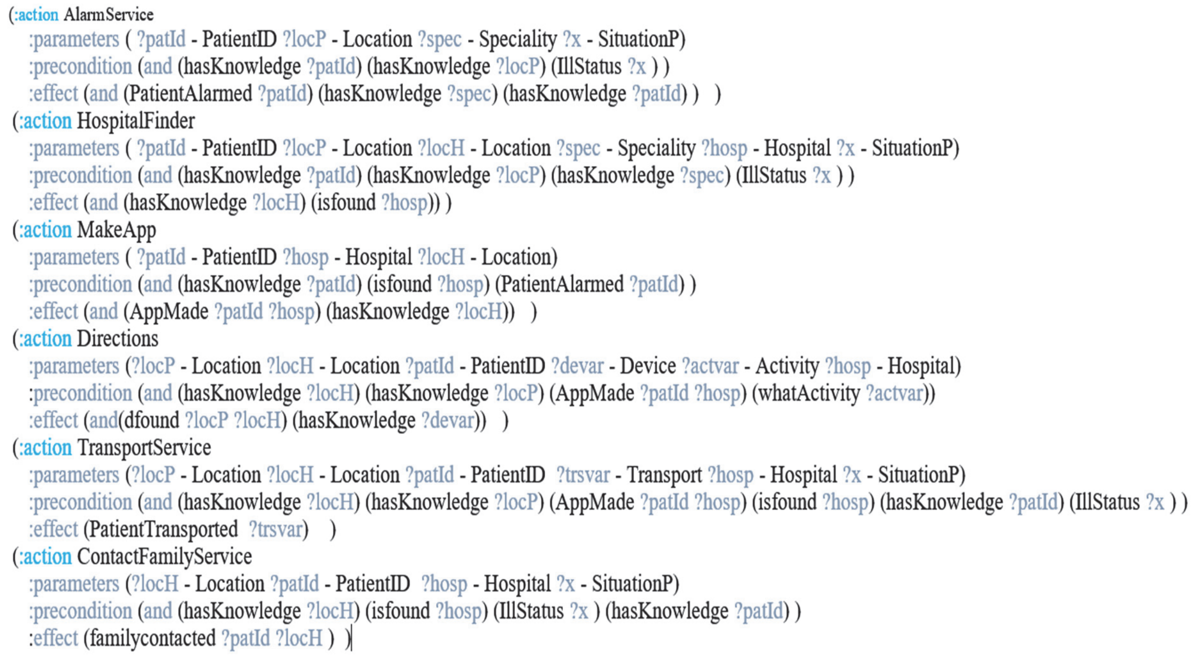
The proposed scenario is about managing and monitoring emergency situations of chronic patients (e.g. diabetes, heart attack, epilepsy, etc.). In this scenario, the proposed approach can be used to plan emergency management flows of actions to be executed.

The composition has to take into account the surrounding context (location, activity, patient situation…etc.). To this end, the proposed approach is able to generate all the planning processes responding to the user's needs.

The assumption is that such cooperating organizations (emergency center, hospitals, doctors, medical transportation) would have made some or all of their emergency management resources or capabilities available as web services.

Each service would have a WSDL (Web Service Description Language) and a CA-OWLS context aware semantic description. CA-OWLS description refers to a general domain ontology that describes the emergency e-health context. To accommodate user requests, the application coordinates atomic, disparate services. For example, some of the services in this scenario include (Figure 10):

- Alarm Service: for notification.
- Hospital Finder: Directory of Hospitals.
- Make App: Service to make appointment.
- Directions service: Navigation service.
- Transport Service: Medical transportation service.
- Contact Family Service: Service to contact families and friends in case of a critical condition.



```
(:action AlarmService
   :parameters ( ?patId - PatientID ?locP - Location ?spec - Speciality ?x - SituationP)
   :precondition (and (hasKnowledge ?patId) (hasKnowledge ?locP) (IllStatus ?x ) )
   :effect (and (PatientAlarmed ?patId) (hasKnowledge ?spec) (hasKnowledge ?patId) )  )
 (:action HospitalFinder
   :parameters ( ?patId - PatientID ?locP - Location ?locH - Location ?spec - Speciality ?hosp - Hospital ?x - SituationP)
   :precondition (and (hasKnowledge ?patId) (hasKnowledge ?locP) (hasKnowledge ?spec) (IllStatus ?x ) )
   :effect (and (hasKnowledge ?locH) (isfound ?hosp)) )
 (:action MakeApp
   :parameters ( ?patId - PatientID ?hosp - Hospital ?locH - Location)
   :precondition (and (hasKnowledge ?patId) (isfound ?hosp) (PatientAlarmed ?patId) )
   :effect (and (AppMade ?patId ?hosp) (hasKnowledge ?locH))  )
 (:action Directions
   :parameters (?locP - Location ?locH - Location ?patId - PatientID ?devar - Device ?actvar - Activity ?hosp - Hospital)
   :precondition (and (hasKnowledge ?locH) (hasKnowledge ?locP) (AppMade ?patId ?hosp) (whatActivity ?actvar))
   :effect (and(dfound ?locP ?locH) (hasKnowledge ?devar))  )
 (:action TransportService
   :parameters (?locP - Location ?locH - Location ?patId - PatientID  ?trsvar - Transport ?hosp - Hospital ?x - SituationP)
   :precondition (and (hasKnowledge ?locH) (hasKnowledge ?locP) (AppMade ?patId ?hosp) (isfound ?hosp) (hasKnowledge ?patId) (IllStatus ?x ) )
   :effect (PatientTransported  ?trsvar)  )
 (:action ContactFamilyService
   :parameters (?locH - Location ?patId - PatientID  ?hosp - Hospital ?x - SituationP)
   :precondition (and (hasKnowledge ?locH) (isfound ?hosp) (IllStatus ?x ) (hasKnowledge ?patId) )
   :effect (familycontacted ?patId ?locH )  )
```

Figure 10. Example of the domain service in PDDL

User requests are enriched with context information. For example, the system takes into account the following context types: user location, user activity, and the computing device in use, provided by a context provider. The following two scenarios are described:

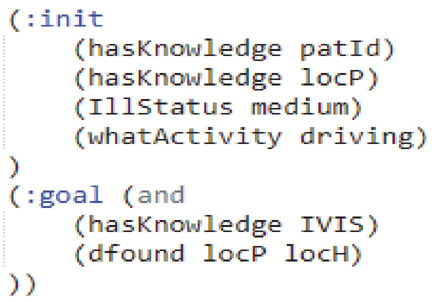
- Scenario 1: A patient equipped with e-health sensors to measure his vital signs (blood sugar level for diabetics), while driving he suddenly he feels pain, his personal agent on his device (car infotainment system CIS) launches a composition request to the emergency center to handle his situation. The resulting composition is made from the following atomic services: notification service Hospital finder a directory of hospitals and clinics, directions service for navigation (Figure 11a).
- Scenario 2: in the second case while the patient is walking near his home and he feels severe pain, he can use smartphone instead of CIS. For this scenario additional services such as a Transport Service and a contact family service are required in the new composition. (Figure 11b).

In both cases, the users have the same goal: managing the patient’s emergency situation. However, the two requests result in the composite services being constructed from different atomic services, because of the different context in which the requests are submitted (Fig 12).

As seen, the result of the planning phase differs as the context changes which makes our approach dynamic. For example if the planner generates a first plan (scenario1) and the monitoring component observes a change in the context before the execution of the composite service, a replanning procedure is launched and a new plan is generated according to the new context. Plans are generated automatically using semantic web services and AI planning techniques.

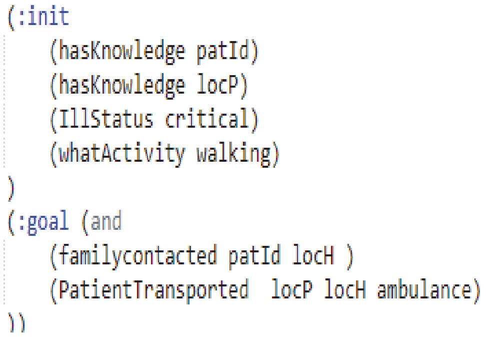
To develop our context aware composition tool, Java as programming language is used with the following technologies:

- Axis2 Eclipse plugin: used to generate the WSDL of web services.
- WSDL2OWLS: was used to generate the OWL-S descriptions from WSDL file.
- Protégé: used to manage ontologies with SWRL plugin to specify preconditions and effects.
- OWL-S editor: used for creating and editing semantic web services.
- PDDL4J: used for parsing PDDL domain and problem.



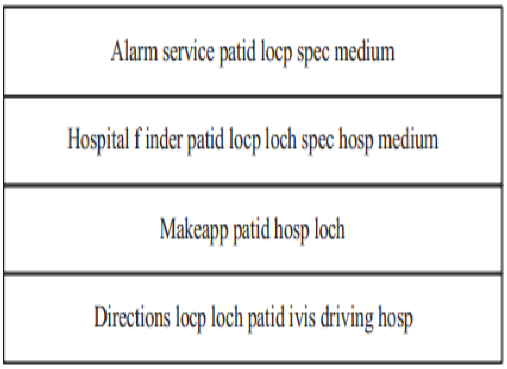
```
(:init
    (hasKnowledge patId)
    (hasKnowledge locP)
    (IllStatus medium)
    (whatActivity driving)
)
(:goal (and
    (hasKnowledge IVIS)
    (dfound locP locH)
))
```

Figure 11a. Scenario 1



```
(:init
    (hasKnowledge patId)
    (hasKnowledge locP)
    (IllStatus critical)
    (whatActivity walking)
)
(:goal (and
    (familycontacted patId locH )
    (PatientTransported  locP locH ambulance)
))
```

Figure 11b. Scenario 2



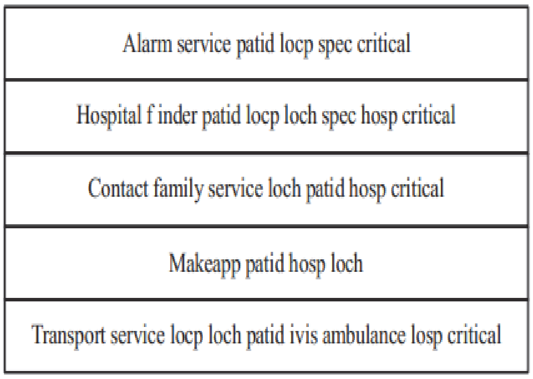
Alarm service patid locp spec medium

Hospital f inder patid locp loch spec hosp medium

Makeapp patid hosp loch

Directions locp loch patid ivis driving hosp

Figure 12a. Generated plan for Scenario 1



Alarm service patid locp spec critical

Hospital f inder patid locp loch spec hosp critical

Contact family service loch patid hosp critical

Makeapp patid hosp loch

Transport service locp loch patid ivis ambulance losp critical

Figure 12b. Generated plan for Scenario 2

4. Related Work

Service composition encompasses all those processes that create added-value services, called composite or aggregated services, from existing services. Lemos et al. (Lemos et al, 2016) proposed an articulated framework for analyzing and comparing Web service composition approaches. There is a distinction between manual/automatic approaches or static/dynamic ones or the combination of the two aspects. Recently different works try to add context awareness to service composition.

Authors (Hatzi et al, 2013) in proposes a system that performs automatic compositions by exploiting service semantic. They use relaxation to have a flexibility in discovering services that will take part of the composition. However, this approach doesn't take context into consideration while discovering and composing services. Da silva et al. (Da silva et al, 2011) presented a QoS-aware approach based on the QoS-model to perform discovery, selection and composition tasks at runtime to enhance user satisfaction and quality of services. By incorporating non-functional parameters such as Response Time and Throughput with the Web services and user request, this approach can select dynamically suitable services in dynamic way. The approach is based on an ontology for semantic description of Web services, which provides interoperability and automation in the Web services tasks. OWLS-XPlan (Klusch et al, 2005) uses the semantic descriptions of atomic web services in OWL-S to derive planning domains and problems, and invokes a planning module called XPlan to generate the composite services. The system is PDDL compliant, as the authors have developed an XML dialect of PDDL called PDDXML. Although the system imports semantic descriptions, the planning module requires exact matching for service inputs and outputs. Generally, the majority of the above approaches don't usually take into account the notion of context awareness.

Li et al. (Li et al, 2011) presented a semantic approach for the composition of services according to the context, aspect oriented programming paradigms are used to weave context into already existed compositions. The approach consider compositions already built which id a drawback of their approach. The AMBIT project aims at providing a base to develop services that are automatically tailored based on the user profile. In their work (Cabri et al, 2016), they explored how services can be composed in a user-aware way. To this purpose, they rely on the SAPERE middleware infrastructure (Castelli et al, 2015), which enables a dynamic and adaptive composition of services based on flexible nature-inspired rules. In their late work Cabri et al. (Cabri2 et al, 2016) proposed to extend the previous approach by considering also semantic techniques and simple collaboration aspects. So, they suggest the definition of a middleware for composing and exploiting services, which exhibits some key features: (i) it considers the profile of the users that exploit the service to choose appropriate services for them, (ii) it exploits techniques of semantic similarity between user and service descriptions to make the choice more effective, and (iii) it enables collaboration among users. The drawback of the approach is the need to automate the composition and recomposition every time the context changes.

Cherif et al. (Cherif et al, 2016) proposed a tool to describe publish and compose services in an adaptive manner, compositions are generated depending on each user

The main advantage of our approach is the use of semantic web to tackle the service composition problem. Modeling context information semantically enables the automation and dynamicity of service composition by using ontologies to model context information and the use of semantic context-aware services. Another advantage of our approach is the use of a three layered architecture to compose and recompose services in case of context changes, therefore adding dynamicity to the composition process.

4. Conclusion

In this paper, we presented our proposal to deal with context aware service composition. First we presented a context ontology and an extension of OWL-S with context parameters in order to make services adaptive to changes in their environment. Next, a three layer architecture is presented to help the creation of composite services and monitor any external event in order to add dynamicity to our approach. AI planning is used alongside semantics in order to automate the composition process. To validate our approach we present an implementation of an e-health scenario to handle emergency cases.

In our future work, we project to tackle the service selection problem to choose the best possible services according to QoS parameters and user preferences.

References

Cabri, G., Leoncini, M., Martoglia, R., & Zambonelli, F. (2016, March). Towards User-aware Service Composition. In International Conference on Nature of Computation and Communication (pp. 11-21). Springer International Publishing. https://doi.org/10.1007/978-3-319-46909-6_2

Castelli, G., Mamei, M., Rosi, A., & Zambonelli, F. (2015). Engineering pervasive service ecosystems: the SAPERE approach. *ACM Transactions on Autonomous and Adaptive Systems (TAAS), 10*(1), 1.

Cherif, S., Cherif, S., Ben Djemaa, R. B., Ben Djemaa, R. B., Amous, I., & Amous, I. (2016). A user-aware approach for describing and publishing context aware composite Web service. International Journal of Pervasive Computing and Communications, 12(2), 174-193. https://doi.org/10.1108/IJPCC-01-2016-0011

Da Silva, E. G., Pires, L. F., & Van Sinderen, M. (2011). Towards runtime discovery, selection and composition of semantic services. *Computer communications, 34*(2), 159-168 https://doi.org/10.1016/j.comcom.2010.04.003

Fikes, R. E., & Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial intelligence, 2*(3-4), 189-208. Jossey-Bass. https://doi.org/10.1016/0004-3702(71)90010-5

Hatzi, O., Vrakas, D., Bassiliades, N., Anagnostopoulos, D., & Vlahavas, I. (2013). The PORSCE II framework: Using AI planning for automated semantic web service composition. The Knowledge Engineering Review, 28(2), 137-156. https://doi.org/10.1017/S0269888912000392

Lemos, A. L., Daniel, F., & Benatallah, B. (2016). Web service composition: a survey of techniques and tools. ACM Computing Surveys (CSUR), 48(3), 33. https://doi.org/10.1145/2831270

Li, L., Liu, D., & Bouguettaya, A. (2011). Semantic based aspect-oriented programming for context-aware web service composition. Information Systems, 36(3), 551-564. https://doi.org/10.1016/j.is.2010.06.003

Martin, D., Burstein, M., Mcdermott, D., Mcilraith, S., Paolucci, M., Sycara, K., & Srinivasan, N. (2007). Bringing semantics to web services with OWL-S. World Wide Web, 10(3), 243-277. https://doi.org/10.1007/s11280-007-0033-

McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., & Wilkins, D. (1998). PDDL-the planning domain definition language. https://doi.org/10.1.1.37.212

Peer, J. (2005). Web service composition as AI planning: a survey (p. 63). Switzerland: University of St. Gallen.

Weiser, M. (1991). The computer for the 21st century. Scientific American, 265(3), 94-104. https://doi.org/10.1038/scientificamerican0991-94

Wang, X. H., Zhang, D. Q., Gu, T., & Pung, H. K. (2004, March). Ontology based context modeling and reasoning using OWL. In Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on (pp. 18-22). IEEE. https://doi.org/10.1109/PERCOMW.2004.1276898

Sheshagiri, M., Sadeh, N., & Gandon, F. (2004, June). Using semantic web services for context-aware mobile applications. In MobiSys 2004 Workshop on Context Awareness.

Rao, J., & Su, X. (2004, July). A survey of automated web service composition methods. In SWSWPC (Vol. 3387, pp. 43-54). https://doi.org/10.1234/12345678

Klusch, M., Gerber, A., & Schmidt, M. (2005, November). Semantic web service composition planning with owls-xplan. In Proceedings of the 1st Int. AAAI Fall Symposium on Agents and the Semantic Web (pp. 55-62).

Cabri2, G., Martoglia, R., & Zambonelli, F. (2016, June). Designing a Collaborative Middleware for Semantic and User-Aware Service Composition. In Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on (pp. 223-228). IEEE.

Copyrights