IJCSI International Journal of Computer Science Issues, Volume 14, Issue 6, November 2017
ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784
www.IJCSI.org https://doi.org/10.20943/01201706.1523

On the use of AI Planning in the Composition of Context-Aware Services

Tarik Fissaa[1], Hatim Guermah[1], Mahmoud El hamlaoui[1], Hatim Hafiddi [1, 2] and Mahmoud Nassar[1]

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

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

Abstract

In context aware ubiquitous systems an important challenge is the composition of multiple services to achieve the user task that cannot be achieved by a single service. With the important amount of available service in such environment, the use of traditional approaches become impractical. By introducing contextual information in the service composition the results of the composite service reacts continuously to context changes. In this paper, we present an overview of a semantic architecture for the discovery and composition of context aware services based on, we also present our context aware semantic service by extending OWL-S with context elements.

Keywords: *Artificial Intelligence Planning, Context Awareness, Context Modeling, Service Composition, Semantic Web Services.*

1. Introduction

The Computing devices are becoming faster, smaller, more widespread and universally connected as the wireless networking and sensing technologies revolution continues. This suggests a vision of systems that are embedded promising the development of Internet of things paradigm [1]. As a result, applications work now in a variety of new settings; for example, embedded in cars or wearable devices. They use context to react and adapt to changes in the computing environment. They are, called context aware systems.

On other side Service Oriented Architecture (SOA) is an architectural style that offers a set of design principles and abstractions for the integration of independent services. The SOA style can be implemented using Web Service (WS) standards and specifications, or other service-based technologies. One of the core principles of Service oriented architectures (SOA) is the idea of assembling services to form a chain by discovering and dynamically composing those multiple existing services to satisfy a user task. Using semantic for the automation of this process is emerging as hot topic facing SOA today [2].

The manual composition of services is complex and susceptible to errors because of the dynamic behavior and flexibility of the context aware services. Thus, to be context-aware composite services need to follow some requirements to resolve the challenges brought by the context-awareness.

AI planning technologies has proven to be useful for the automation of services composition. By treating services as an action, planners do various sorts of reasoning about how to combine services into a plan responding to the user goal. AI planning based algorithms try to find a feasible composition solution through search of possible services to accomplish a specific task.

For the above reasons, context-aware service development can benefit from semantic web services and AI planning techniques. The purpose of Semantic Web Services is to use semantic specification to automate the discovery, invocation, and composition Web services. The Ontology Web Language for services (OWL-S) [3] is the most direct outcome for describing Web services in the semantic web.

In this work, we aim to propose in first stage an overview of an architecture for the automation of context aware services composition. Our major contribution in this paper is the proposition of an extension to OWL-S with context elements to take advantage of the context awareness, and especially we propose a tool for automatic services composition using AI planning as recommended in semantic web services OWL-S specifications.

The remainder of this paper is structured as follows. Next we present some related work in context aware service composition approaches. In section 3 we presents our proposal for context modeling and context aware semantic services specifications. In section 4 we give an overview of the proposed architecture and his different layers. In section 5 we present some results by applying the proposed architecture in the e-health scenario. The final section gives rise to some concluding remarks and plans for future works.

2. E-health Motivating Scenario

In this scenario, the underlying idea is to allow elderly people, chronic patients to stay at home and to benefit from a remote and automated medical supervision. Let's take the case of a patient with a cardiac arrhythmia. The patient's situation requires monitoring to detect crises or problems that may be caused by the increase or decrease in his heart rate. Thus in the case of a problem the patient should be informed of a critical condition, and take the necessary precautions. The role of the context-aware system is to find the nearest caregiver (nurse or available physician) to help the patient depending on the severity of his condition. In parallel, depending on the current situation of the patient, the system must find an ambulance to transport the patient to the hospital. The aim is to provide the patient with both higher levels of security and independence that allows him to live a normal life despite his illness. This e-health scenario highlights the fundamental challenges for the composition of context aware services. Having regard to the need to use the patient context, the e-health system should be context aware. Therefore, different types of services can be defined (see Figure 1).

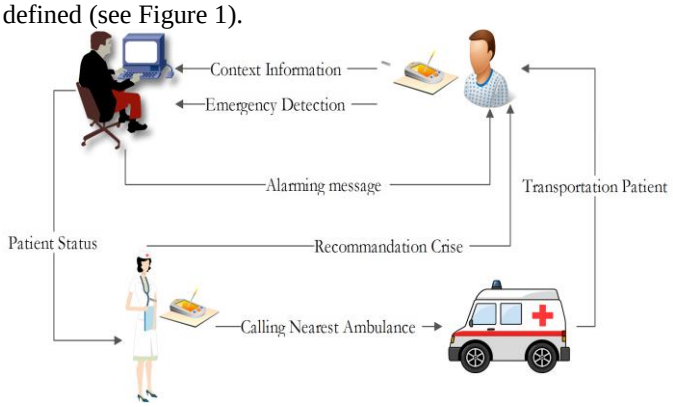

Fig. 1 E-health scenario

3. Related work

3.1 Context Modeling

The notion of 'context' is vague to define, many researchers working on context had given a variety of different definitions. However, a universally accepted definition is yet to be agreed. Schilit in [5] defined context to be: location, identities of nearby people, objects and changes to these objects. Other definitions had been proposed, Brezillon [6] define context as an information that characterizes the interactions between humans, applications and the environment. Dey et al. [7] discuss that the important aspects of context cannot be enumerated, as they differ from situation to situation and depend on the purpose of the application, furthermore they formally defined context as:"... any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves."

In our work we will adopt this definition because it remains the most generic.

Several approaches for context modeling:

i. Key-value models:

The model of key-value pairs is the most simple data structure for modeling contextual information. Already Schilit et al. [8] used key-value pairs to model the context by providing the value of a context information to an application as an environment variable. Key-value pairs are easy to manage, but lack capabilities for sophisticated structuring.

ii. Markup Scheme Models:

Common to all markup scheme modeling approaches is a hierarchical data structure consisting of markup tags with attributes and content. Some of them are defined as extension to the Composite Capabilities/Preferences Profile (CC/PP) [9] which have the expressiveness reachable by RDF/S and a XML serialization.

iii. Graphical Models:

A very well-known general purpose modeling instrument is the Unified Modeling Language (UML) which has a strong graphical component. Due to its generic structure, UML is also appropriate to model the context. This is shown for instance Henricksen et al., in [10] which is a context extension to the Object-Role Modeling (ORM) approach according some contextual classification and description properties.

iv. Object Oriented Models:

Common to object oriented context modeling approaches is the intention to employ the main benefits of any object oriented approach - namely encapsulation and reusability – to cover parts of the problems arising from the dynamics of the context in ubiquitous environments

v. Logic Based Models:

A logic defines the conditions on which a concluding expression or fact may be derived (a process known as reasoning) from a set of other expressions or facts. In a logic based context model, the context is consequently

defined as facts, expressions and rules. Usually contextual information is added to, updated in and deleted from a logic based system in terms of facts or inferred from the rules in the system respectively. Common to all logic based models is a high degree of formality.

vi. Ontology Based Models:

As the context may be considered as specific kind of knowledge, it can be modeled as ontology. Ontologies are a very promising instrument for modeling contextual information due to their high and formal expressiveness and the possibilities for applying ontology reasoning techniques. There are several ontology based approaches among them CONON (stands for CONtext Ontology) [11], CoBrA-ONT (Context Broker Architecture ONTology) [12].

Ontologies have proved to be the most suitable model for representing and reasoning on context information for the following reasons:

- ontologies enable knowledge sharing in open, dynamic systems;
- ontologies with well-defined declarative semantics allow efficient reasoning on context information;
- Ontologies enable service interoperability.

3.2 Context-aware service composition approaches

Service composition encompasses all those processes that create added-value services, called composite or aggregated services, from existing services. Lemos et al. [13] 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.

Hatzi [14] presented PORSCE II, an integrated system that performs automatic semantic web service composition exploiting AI planning. The main advantage of the proposed framework is the extended utilization of semantic information, in order to perform planning under semantic awareness and relaxation. In our point of view this is an interesting approach to dealing with semantic composition, however the context dimension is ignored and not taken into account.

OWLS-XPlan [15] 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. [16] presented an approach to support context-aware semantic service composition, by weaving context aspects, defined by means of ontology concepts, within plain compositions. Weaving is performed statically, before starting the execution of the main service. However, they don't deal with context modeling and reasoning thus automatic composition is not considered. Mrissa et al. [17] tried to solve Input/output compatibility issues between services within a composition of services by integrating the notion of context and service mediator. The used concept of context does not take into account the underlying definitions of the context relative to ubiquitous environments and the fact of inserting a mediator service within each pair of services in the composition process may, from our point of view, significantly increase response time and system performance.

Cherif et al. [18] proposed a framework that supports the development, the description and the publication of Adaptive Service Composition (ASC). The objective is to offer relevant and suitable information depending on a user's particular profile. Therefore, developers have to integrate software facilities dealing with context characteristics. Then, context annotations must be used to extend BPEL descriptions.

Our approach is in line with the previous work in our research team [19] [20].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.

4. Context and context aware service modeling

4.1 Ontology based Context Modeler

To model context in generic and abstract way, we propose a metamodel that defines the context and its sub context, context property, validity, and specifications of each context property (see Fig.2).

This metamodel is based on the following specifications:

- A context decomposes into sub contexts;
- A sub context can be, recursively, decomposed into categories for its structuring;
- A context, a sub context and a category are constituted of context properties;
- A context property is gathered by sensors: SensedCtxProperty, or derived from other context

properties: DerivedCtxProperty, or stored in the database: StoredCtxProperty.

- Each property has a context validity.
- A derivedCtxProperty is obtained by derivation from a set of properties based on derivationFunction.
- A StoredCtxProperty is obtained through the recovery features of stored properties (e.g.recoveryFunction)
- A sensedCxtProperty is obtained and characterized by the context source SourceCxtProperty and the acceptance value of context property QualityofProperty
- Each sub context has a specific type of access.

To illustrate our metamodel, let's project it on the case of figure of the E-health system. The context for this system in particular and context-aware computing in general is composed mainly of the following sub contexts:

- Environment: represent user's location, time...etc.
- Medical Information: contains user's health properties (in our case Blood Sugar, HRV, and Blood pressure).
- User: contains properties describing the user (Blood group, age, preference).
- Device: contains parameters that describe the entity Device (e.g. medicals device, mobile phones, PDA...etc.).

4.2 Context metamodel to OWL transformation

Models transformations provide a mechanism for automatically creating or updating target models based on information contained in existing source models. Formally, a simple model transformation has to define the way for generating a model Mb, conforming to a metamodel MMb, from a model Ma conforming to a metamodel MMa. In our case, the source metamodel corresponds to the context metamodel and the target metamodel corresponds to the OWL metamodel. As a result, an OWL ontology based on our context model [21] was obtained.

The resulting ontology consists of a set of classes, individuals, properties and relations that describe the various context properties (Fig 3). This OWL ontology can be used for reasoning, discovering and composing services.

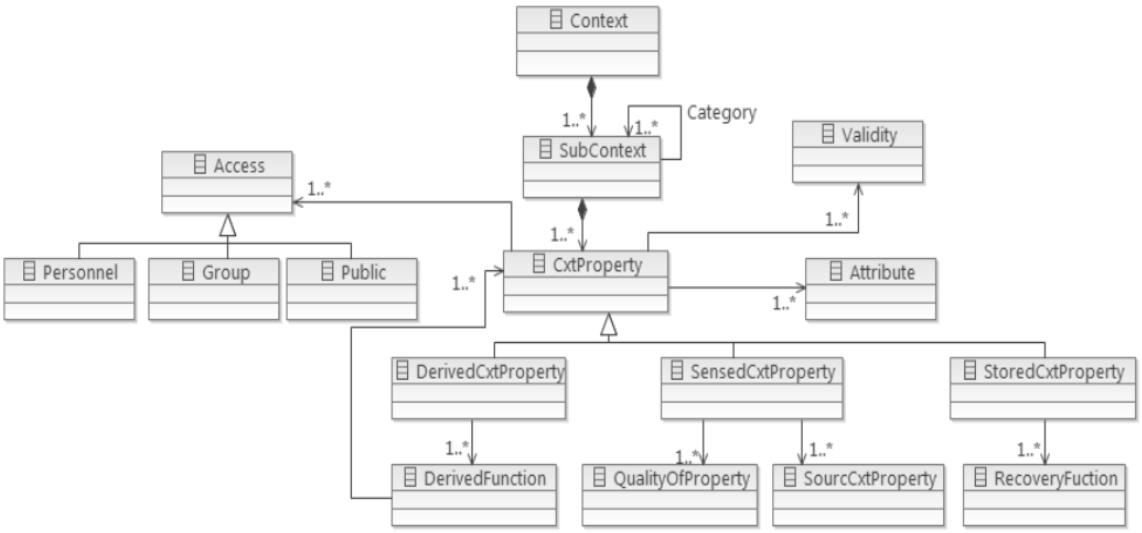

Fig. 2. Context Metamodel

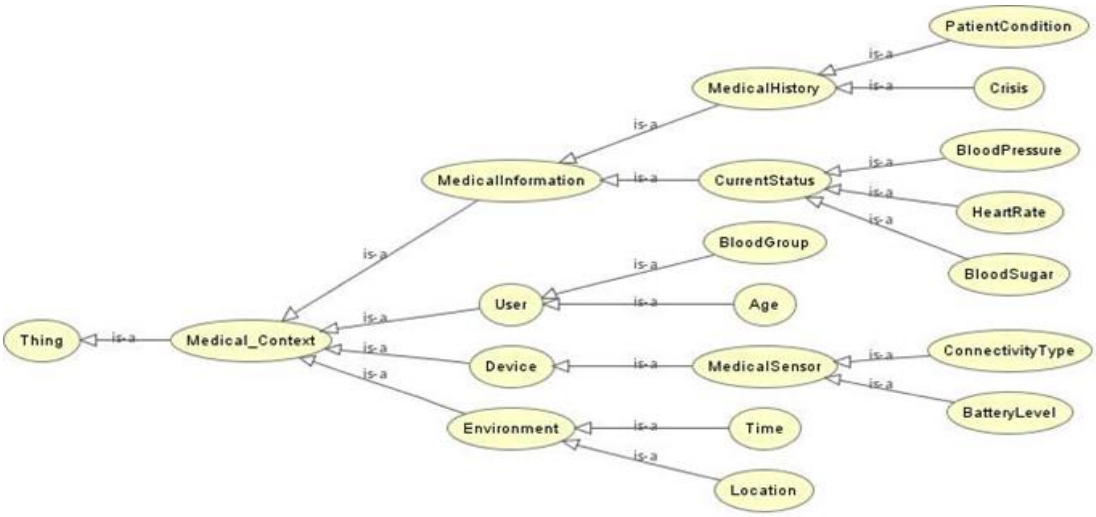

Fig. 3 Excerpt of the OWL context ontology in Protégé editor

4.3 Context aware service description

OWL-S [3] 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.

Fig.4 illustrates our Semantic Context Aware Service (CA-OWLS). The CA-OWLS is based on the following specifications:

- The SCAS has a ServiceModel, ServiceProfile and ServiceGrounding.
- The service model can be viewed as 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 certain external pre-condition to be satisfied to execute the process.
- Adaptation Effect: The execution of the service may result in certain external effects.
- QoS parameters for service selection issue.

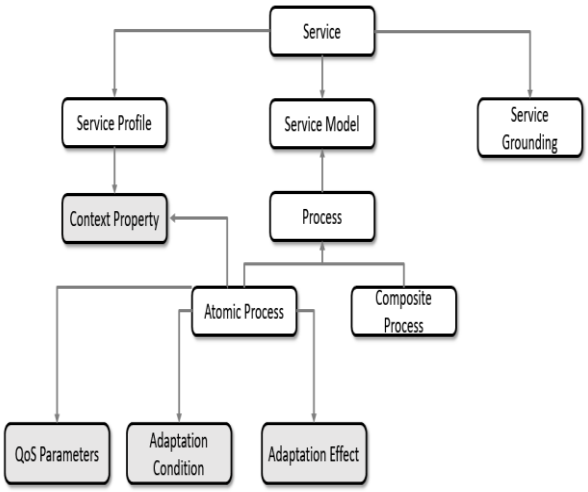

Fig. 4 OWL-S Extension with context elements (CA-OWLS)

5. Architecture Overview

To address these limitations discussed above, we propose a novel architecture for service composition in context aware environment. This approach is grounded in the process of dynamically discovering, composing and coordinating atomic or already composite services, based on the current context of a user. Thus, services are continuously recomposed as response to context changes, enabling the automated development and adaptation of context aware applications. Furthermore, the proposed architecture

embodies a composition request management system, which attempts to reformulate composition requests into alternative ones depending on context changes.
Our approach for context-aware service Composition is based on an integrated environment based on the synergy between Semantic Web Ontology and Context Awareness. Fig.5 presents an overview of all system's component aimed at providing personalized services according to the user context. There are the main components that we distinguish in the architecture.

Fig. 5. The three layers of the proposed architecture

5.1 Context goal request

The context goal manager layer assembles and, if necessary, modifies a composition request. Fig. 6 shows the structure of this layer that represents the first layer of the composition architecture. The composition request is an entry point to the composition process. It specifies the user's task and consists of two parts. The first one is a description of the core user request. The second part contains contextual parameters. Such contextual parameters further personalize the composition request.

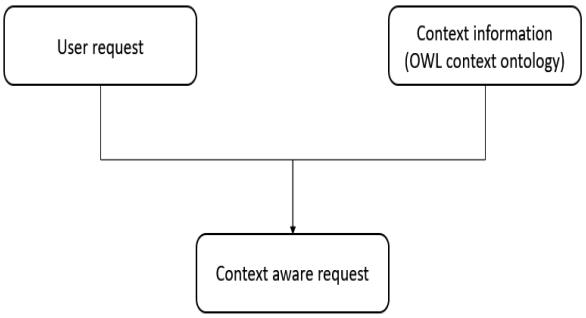

Fig. 6 The context aware request

5.2 Context aware service composition architecture

To automate the composition of context-aware services, we have to resort a strategy for composing and adapting context aware services. Figure 6 shows the process of composing our semantic context aware services using AI planning. The relevant extended OWL-S services description is acquired by semantic services discovery. The key features of the composer are:

- Service discovery: discovering appropriate service to be composed.
- Translate the Web services description of domain and problem written in OWL-S to the planning problems described in PDDL[22]
- Input the result generated by the converting process into AI planner.
- Translate back the actions sequence that AI planner got in the precedent step into a composite Web service.

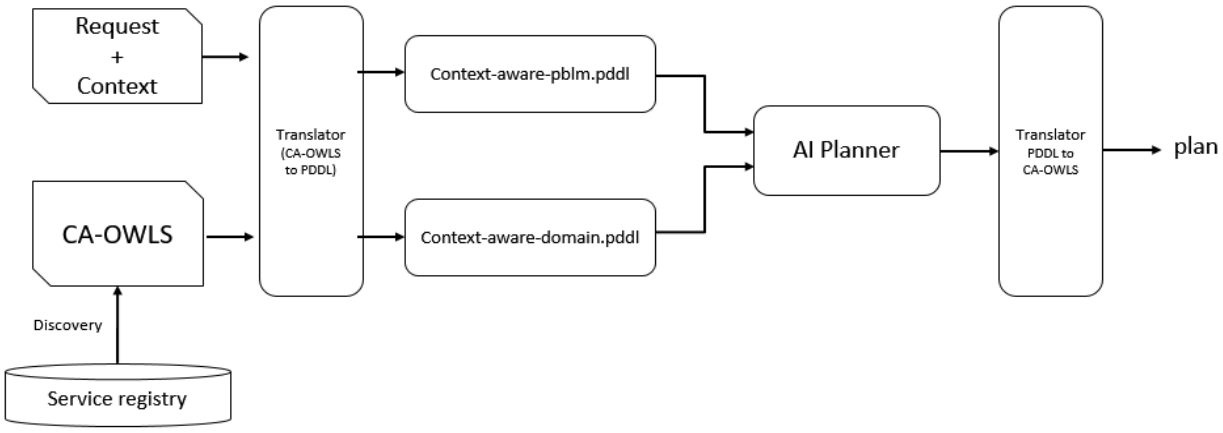

Fig. 7 Context-Aware Service Composition

5.2.1 Service discovery

The inputs of our matching component are: the users query enriched with context elements, a set of advertised services (i.e. service descriptions), and the matchmaking module. A base algorithm for service signature matching has been proposed by Paolucci et al. in [23]. This algorithm allows matching a requested capability, described as a set of provided inputs and required outputs, with a number of advertised capabilities, described each as a set of required inputs and provided outputs. Inputs and outputs are semantically defined using ontology concepts. Our approach also supports the matching of context properties and provides a means to rate services according it. To rate how relevant particular match between R and S is, we use the number of service properties (i.e. type, inputs, outputs and contextual attributes). We define four levels of matching between a provided and a required ontology concept. The four matching levels are:

- Exact: if the concepts are equivalent or if the required concept is a direct subclass of the provided one
- Plug in: if the provided concept subsumes the required one.
- Subsumes: if the required concept subsumes the provided one.
- Fail: if there is no subsumption relation between the two concepts

5.2.2 Translator CA-OWLS to PDDL

The Planning Domain Definition Language (PDDL) proposed by Mcdermott et al. [22] and inspired by STRIPS [24] (Stanford Research Institute Planning System), has become a standard language for describing planning domains. PDDL uses precondition and effects to describe the applicability and effects of actions. The language has been widely accepted in the AI planning community, since it standardizes the domain description and the problem description in planning research. Planning tasks specified in PDDL are separated into two files:

- The domain description file contains the definition of all types, predicates and actions, whereas
- The problem description file includes all objects, the initial state, and the goal state.

The role of the translator is to convert the set of CA-OWLS descriptions and the context goal ontology into a domain and problem planning respectively. In summary, only the essential of the translation is described.

For the creation of PDDL actions, CA-OWLS web services description is used. Each CA-OWLS description consists of three main parts: profile, process and grounding. A CA-OWLS process is described by (inputs, outputs, preconditions, effects, context parameters, Adaptation conditions and Adaptation effects). From this description a PDDL action parameters and action body is created.

Input/output and context parameters in the service description are converted into additional service preconditions and effects by using a specific predicate (hasKnowledge).

5.2.3 Planning process

Planning focuses on selecting suitable actions and ordering them in an appropriate sequence so as to achieve some goal. In general, a classical AI planning for service composition problem can be formalized as a quintuple < S; S0; G; A; T>, where:

- S is the set of all possible states of the world.
- $S_0 \subset S$ denotes the initial state of the world.
- $G \subset S$ denotes the goal state of the world the planning system attempts to reach.
- A is the set of actions the planner can perform in attempt to reach a desire goal (web services in terms of service composition).
- The translation relation $T \subseteq SxAxS$ defines the precondition and effects for the execution of each action.

In terms of Web services, S0 and G represent the initial state and the goal state respectively, specified by the service requestors. A is a set of available services and G denotes the current states of each service. So far we have presented available Web Services and initial and goal state of the problem domain in PDDL. The domain and problem files can be sent to any PDDL based planner to generate a valid composition plan. The generated plan provides the sequence of the context aware service composition. One of the most prominent planning algorithms is Graphplan [25] it consists of two interleaved phases: a forward phase, where a data structure called *planning-graph* is incrementally extended, and a backward phase where the planning-graph is searched to extract a valid plan. The planning graph can be created in a polynomial time, with respect to the size of the problem domain, while the search phase has an exponential complexity in the worst case.

After the acquisition of solutions, a reverse translation process has to take place, in order to provide the resulting composite web service to the original OWLS standard and the initial web services domain. This reverse translation accommodates composite service deployment and execution monitoring.

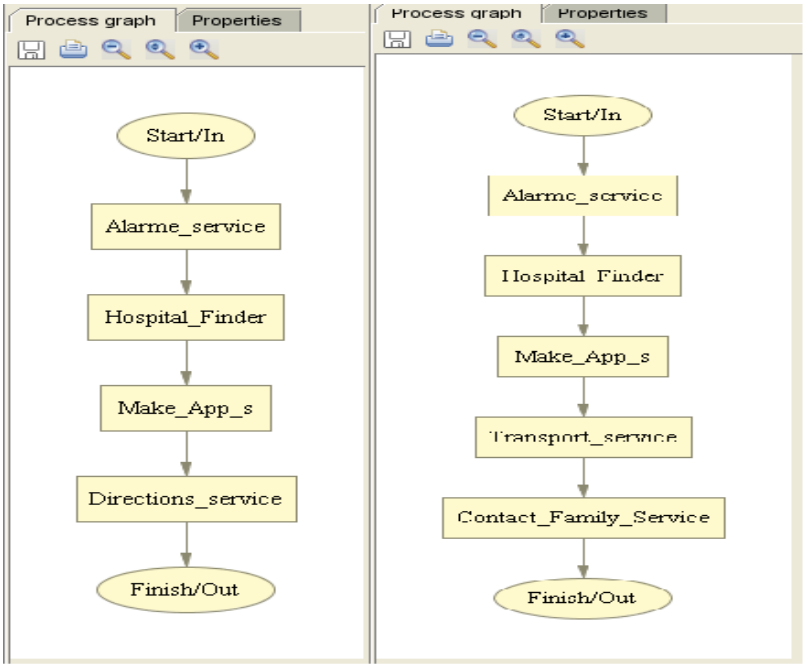

Figure 8 resulting composition

5.3 Execution and monitoring

The Execution Engine provides the run-time environment in which services can be executed. It invokes scheduled services as specified in the deployable service description.
The Monitoring is bound to the Execution Engine to track changes in the run-time environment, service performance and composition request status. The Monitoring Engine verifies the service preconditions before being invoked by the Execution Engine. During the service lifetime it observes changes in the environment and reacts to context changes. Finally, once the service completes its operation the Monitoring Engine verifies service effects, against the expected outcomes.

6. Results and Discussion

In the previous discussed e-health motivating scenario, the objective is to give elderly people suffering from chronic disease (e.g. diabetes, heart attack, epilepsy, etc.) a way to manage and monitor their situations and to benefit from a remote and automated medical supervision.
In the previous discussed e-health motivating scenario, the objective is to give elderly people suffering from chronic disease (e.g. diabetes, heart attack, epilepsy, etc.) a way to manage and monitor their situations and to benefit from a remote and automated medical supervision. Each service would have a semantic context aware description by using our extension CA-OWLS. To be used by a planner the set of CA-OWLS is converted to a domain file in PDDL, the context aware request is transformed into a problem file. Some of the services in this scenario include: an Alarm Service for notification, Hospital Finder service, Transport Service…etc.
Let's take the case when the A patient equipped with e-health sensors while driving he feels pain, his personal agent on his device (his vehicle device) 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.
The second case while the patient is walking he feels severe pain, he can use smartphone. For this scenario additional services such as a Transport Service and a contact family service are required in the new composition.
The figure 8 show a graphical representing in Protégé of the resulting composition for each case.
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

7. Conclusion

In this paper a context aware approach for service composition using AI Planning was presented. We presented our context modeler following MDE to generate an OWL context ontology. Next step, we used this ontology along with a context extension of semantic web service OWL-S to automate the composition of services according to user context. Thus an architecture that compose those services is presented, given a description of user task and a set of semantic context aware services.
We project to provide an applicative layer of our tool for service composition in order to automate the whole process of the composition. We also plan to evaluate more composition mechanisms such as heuristics and other problem solving algorithms.

References

[1] Perera, C., Zaslavsky, A., Christen, P., & Georgakopoulos, D. (2014). Context aware computing for the internet of things: A survey. IEEE Communications Surveys & Tutorials, 16(1), 414-454.
[2] Rao, J., & Su, X. (2004, July). A survey of automated web service composition methods. In SWSWPC (Vol. 3387, pp. 43-54).
[3] 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.
[4] Peer, J. (2005). Web service composition as AI planning: a survey (p. 63). Switzerland: University of St. Gallen.
[5] Schilit, B., Adams, N., & Want, R. (1994, December). Context-aware computing applications. In Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on (pp. 85-90). IEEE.
[6] Brézillon, P. (2003). Focusing on context in human-centered computing. IEEE Intelligent Systems, 18(3), 62-66.
[7] Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., & Steggles, P. (1999, September). Towards a better understanding of context and context-awareness. In International Symposium on Handheld and Ubiquitous Computing (pp. 304-307). Springer Berlin Heidelberg.
[8] Schilit, B. N., & Theimer, M. M. (1994). Disseminating active map information to mobile hosts. IEEE network, 8(5), 22-32.
[9] Nilsson, M., Hjelm, J., & Ohto, H. (2000). Composite capabilities/preference profiles: Requirements and architecture. W3C Working Draft, 21, 2-28.
[10] Henricksen, K., Indulska, J., & Rakotonirainy, A. (2003). Generating context management infrastructure from high-level context models. In In 4th International Conference on Mobile Data Management (MDM)-Industrial Track.
[11] 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).
[12] Chen, H., Finin, T., & Joshi, A. (2003). An ontology for context-aware pervasive computing environments. The knowledge engineering review, 18(03), 197-207.
[13] Lemos, A. L., Daniel, F., & Benatallah, B. (2016). Web service composition: a survey of techniques and tools. ACM Computing Surveys (CSUR), 48(3), 33.
[14] 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.
[15] 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).
[16] Li, L., Liu, D., & Bouguettaya, A. (2011). Semantic based aspect-oriented programming for context-aware web service composition. Information Systems, 36(3), 551-564.
[17] Mrissa, M., Ghedira, C., Benslimane, D., & Maamar, Z. (2006, November). A context model for semantic mediation in web services composition. In International Conference on Conceptual Modeling (pp. 12-25). Springer Berlin Heidelberg.
[18] 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.
[19] Hatim Hafiddi, Hicham Baidouri, Mahmoud Nassar, Abdelaziz Kriouile Context-Awareness for Service Oriented SystemsIJCSI International Journal of Computer Science Issues, Vol. 9, Issue 5, No 2, 2012
[20] Baidouri, H., Hafiddi, H., Nassar, M., & Kriouile, A. (2015). Enabling Context-Awareness for Dynamic Service Composition. International Journal of Advanced Pervasive and Ubiquitous Computing (IJAPUC), 7(1), 17-29.
[21] Fissaa, T., Guermah, H., Hafiddi, H., Nassar, M., & Kriouile, A. (2013, December). Ontology Based Context Modeler for Context-Aware Systems (Short paper). In Service-Oriented Computing and Applications (SOCA), 2013 IEEE 6th International Conference on (pp. 43-47). IEEE.
[22] McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., ... & Wilkins, D. (1998). PDDL-the planning domain definition language.
[23] Sycara, K., Paolucci, M., Ankolekar, A., & Srinivasan, N. (2003). Automated discovery, interaction and composition of semantic web services. Web Semantics: Science, Services and Agents on the World Wide Web, 1(1), 27-46.
[24] 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.
[25] Blum, A. L., & Furst, M. L. (1997). Fast planning through planning graph analysis. *Artificial intelligence*, *90*(1), 281-300.