Automated web service composition with knowledge approach

Zoltán Ďurčík *
Department of Cybernetics and Artificial Intelligence
Faculty of Electrical Engineering and Informatics
Technical University in Košice
Letná 9, 042 00 Košice, Slovakia
zoltan.durcik@tuke.sk

Abstract

The topic of this thesis is web service composition with using planning methods and knowledge approach. Web services are programs located on computer networks (mainly on internet). Each web service provides certain functionality (e.g. translation between two languages). If it is not possible fulfill request by one web service (e.g. it's needed translate word from English to Spanish but there is not available service, which directly translates words between these two languages), web services composition can help here (e.g. there is one web service to translate English word to German, and second service, which translates German word to Spanish). In presented thesis were introduced own approaches for handling this problems, which outgoing from existing methods and system for automated web service composition. These approaches are based on using *knowledge representation*, either on ontological web service description (OWL-S) and other ontological description, e.g. description of domain (OWL - Ontology Web Language). The composition is performed by *artificial intelligence planning methods*. For planning was choices *PDDL* language. In this work is introduced methods for transformation ontologies into PDDL planning task. Beside this is there proposed complete web service composition system, which was also partially implemented.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Plan execution, formation, and generation*; H.3 [**Information Storage and Retrieval**]: On-line Information Services—*Web-based services*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representation languages, Semantic networks*

Keywords

web service, composition, planning, planner, semantic, knowledge, ontology, OWl, OWL-S, PDDL, transformation

1. Introduction

Web services (WS) are distributed programs located on networks (most frequently on internet) and using by standard protocols (most frequently by HTTP). The concept of WS was introduced by major IT corporation as Microsoft, IBM and Sun and was proposed as alternative to object-oriented distributed standards as COBRA[1] and Java RMI [2]. There are two main properties of WS: they must be *self-descriptive* and they must be *interoperable* together regardless to environment (program language), in which they was created. The self-describing of WS is allowed by service descriptions, which are available together with services. WSDL (Web Service Description Language) is one of the most used language for WS descriptions. The independence WS from environment is given by theirs communication style. They communicate by structuralized messages. Majority of WS using for communication XML messages, which utilize SOAP (Simple Object Access Protocol) standard. Beside SOAP oriented WS exist also other approaches e.g. REST. Submitted work but dealt only with SOAP WS.

WS may be presented as standards programs, which provide some functionality (e.g. service translates word from one language to another). The information about WS inputs and outputs is located in WSDL description. Beside this information is in WSDL descriptions described also the way of interaction with the specific service. If there isn't possible fulfill user request by one WS, then occurs the problem of web service composition. There may exist WS, which composition is able to fulfill user request. The goal of WS composition is to produce workflow and dataflow created from available WS. With execution this workflow is attained required result. Given workflow may

*Recommended by thesis supervisor:
Assoc. Prof. Ján Paralič
Defended at Faculty of Electrical Engineering and Informatics, Technical University in Košice on October 26, 2010.

Ďurčík, Z. Automated web service composition with knowledge approach. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 2, No. 2 (2010) 30-37

[1]Common Object Request Broker Architecture, http://www.cobra.org.
[2]Java: Remote Method Invocation: http://java.sun.com/j2se/1.4.2/docs/guide/rmi/index.html

be represented e.g. by plan, where particular WS may be connecting by defined constructs (e.g. if then else, cycles etc.).

There exist a various methods and system, which dealt with automated web services composition (AWSC). Several of this methods and systems using artificial intelligence planning methods for AWSC (e.g. [13][18][17]). The goal of this thesis was analyse the possibility of using knowledge approaches together with this planning methods for fulfillment AWSC problem and provide own system proposal for this possibility. More detailed objectives description can be seen in next subsection.

1.1 Thesis objectives

The work deal with AWSC and is especially focused on the composition problem definition. For the primary composition problem definition are used knowledge approaches (OWL and OWL-S ontologies), which in next step must be translated into planning task. The objectives of this thesis were defined as follows:

- *Analyze actual state in the web service area and web service composition domain.*
  The goal of this step is present a survey of WS, standards and technologies related with WS from a WS composition perspective. Next goal is introduce general WS composition system, and describe particular parts of this system.
- *Analyze available planning methods and approaches in term of their usability for web service composition.*
  The goal is an analyze of planning methods in terms of WS composition requirements, next provide a survey of most widely used AI planning methods, as well as existing proposals and systems for WS composition. Further the aim is delimiting a suitable planning language for WS composition.
- *Propose own solution for WS composition with utilize knowledge approaches.*
  In regards to WS composition problem, planning methods and existing systems analysis, the goal is to provide own system for WS composition, which should use a knowledge approaches for primary WS composition problem description.
- *Implement and experimental verify a functionality of proposed system for knowledge base AWSC.*
  The goal of this step is an implementation WS composition system, with focus on semantic problem description into planning task transformation, and experimental verification.

Following these objectives is work dividing in two main logical ports. First parts provides theoretical base in the AWSC area (sec. 2, 3, and 4). Second part (including sec. 5 and 6) contains own system proposal with a main part of this work, namely with a transformation process from external composition problem specification to internal specification (see fig. 1).

2. WS standards and technologies

Web services are distributed programs located on networks (most frequently on internet) and used by standards protocols (most frequently by HTTP protocol).

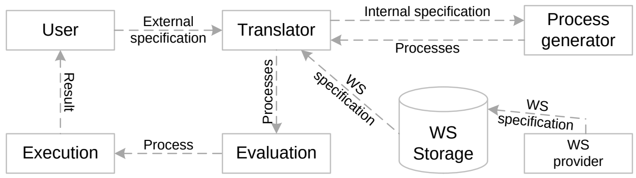

Figure 1: Automated WS composition framework

Among most important standards and technologies, which related with WS compositions issue, belong:

- *WSDL* (Web Service Description Language)[2] is the most used language for WS description. It makes possible describe the offer of available operation in WS, parameters of these operations, a way of communication with WS etc.
- *SOAP* (Simple Object Access Protocol)[3] is XML based protocol. It serves to interaction with WS and to information exchange among WS over network interface by using XML messages. It is platform and programming language independent. It's mean that it was proposed with the aim of independent of concrete programming language model.
- *UDDI* (Universal Description, Discovery and Integration) is a standard to registration, categorization and discovery of WS. Methods of working with UDDI evoke a catalogue, in which is located information about WS providers and about provided WS. WS, which are registered in UDDI, must work with standards as WSDL and SOAP. Each WS in UDDI must have assigned explicit WSDL document.
- *OWL* (Ontology Web Language)[4] is the language designed for ontology description on internet. It belongs among languages, which serve to knowledge representation, and it is approved by W3C[3] consortium. In a computer science ontology presents formal knowledge representation by a set of concepts and a relationship among these concepts in some domain. OWL is semantic language for publishing and sharing ontologies.
- *OWL-S* (Semantic Markup for Web Services)[5] may be characterized as ontologically descriptive language for WS. The base WS descriptive languages (as e.g. WSDL) are in terms of semantic web insufficient (mainly in terms of automation some activity on internet). But with incoming ontologies was showed, that there is a possibility describes WS by ontologies, and connect these descriptions with existing ontologies. These descriptive ontologies for WS were called OWL-S (for reason that they are OWL ontologies and are used for service describing). Mainly motivational task, which suggests creation of OWL-S, was: automated WS discovery, automated WS invocation and automated WS composition.

3. Automated web service composition

Automated web service composition (AWSC) presents a manner of processes creation, which are represented by workflows. This workflows consist of available WS, which

[3]World Wide Web Consortium: http://www.w3.org/

Table 1: Comparison of selected systems for WS composition

System	Process generator technique	WS standards support	Internal representation of problems	System type	Dynamic interference with system
WS composition - PROLOG	Logical programming	OWL-S, WSDL	Logical programming programs	semiautomatic composition and collaboration with user	no
SHOP2 system	HTN planning	OWL-S, WSDL	SHOP2 domain	automatic composition	no
OWLSXplan	Extended forward planning FF and HTN planning	OWL-S, WSDL, OWL	PDDL 2.1	automatic composition	yes, OWLSXplan version 2
WSPlan	Unspecified AI planning methods	WSDL	PDDL 1.2	automatic composition with manually WS choice	no
SEMCO-WS	Petri Nets	WSDL,OWL-S, OWL	GWorkflowDL	semiautomatic composition by workflow	no

are needed for users request solutions. On Fig.1 is pictured general system design for AWSC. This is one of simplest system architecture, on which may be explained WS composition process. Among particular parts belong:

- *WS storage* - serves on storage WS received from WS providers.
- *Translator* - serves to information processing received from users and from WS storage. By using this information is next defined external specification of WS composition problem. The translator performs a process of transformation between external and internal specification of WS composition problem. The external definition may use for problem definition e.g. semantic web standard as OWL ontologies and OWL-S ontological WS descriptions. The internal specification is selected in retard to type of process generator (e.g. if the problem generator is a planner, than the internal specification may be represented by using specific planning language - for example PDDL, Prolog etc.).
- *Process Generator* - following user request, which is translated into internal specification, is in this part realized process creation. At the end of this is providing a set of atomic WS with workflow and dataflow among these WS.
- *Evaluator* - evaluation occurs in case, if there is available a lot of processes, which satisfy user request.
- *Executor* - execution of selected WS in order given by process generator. The result of this execution is next provide to user.

4. Planning

To solve the WS composition problem may be used various methods, e.g. Petri Nets [23], situation calculus[15] or an artificial intelligence planning methods [12][19][14]. Just then the last mentioned AI planning methods are considered as the best choice to AWSC process generator. Planning task in general consists from following parts [27]: the description of available actions (definite in some formal language - domain theory), the initial state description of planning and the goal (final) state description. Planning task may be represented as some world model and may be write as pentad $<S,So,G,A,\Gamma>$[13], where:

- S represented a set of all possible states in given world model,
- So is a subset of S, and represent initial state
- G is a subset of S, and represent goal state,
- A is a set of all available actions, from which each change the world state by passing its from one state to another,
- relation Γ is a subset of $S \times A \times S$ and define preconditions and effects for each action.

Among most used methods for AI planning belongs e.g.:

- *space-state planning*[7],
- *graph oriented planning*[25],
- planning by using *hierarchical nets*[17],
- planning by using *logical programming*[16].

4.1 PDDL

In case of there is used planner as process generator in AWSC system, is needed to choose formal language, which will be used for planning task definition. As one of most suitable language for this choice may be consider *PDDL* language (*Planning Domain Definition Language*). Language PDDL arose out of effort planning domain and planning problem description standardization. It arose for international planning competition requirements ICP

[4] and this language is among planning community greatly popular. There exist various planners, which use PDDL as formal planning language, and may be used in AWSC system (e.g. [10][11]). Planning task is in PDDL divide into two sets:

- planning domain definition and
- planning problem definition in regard to planning domain.

Planning domain definition consists from domain name definition, next from types, which may be used in domain and in corresponding planning problem, from predicates definition and from sets of available actions. A planning problem definition binds to some planning domain. Therefore is necessary in PDDL problem definition defining name of this domain. Next this definition contains concrete objects, initial and goal state.

4.2 Existing proposals and systems

There exist several proposal and system for (semi-)automated web service composition. In table 1 are presented and compared selected systems for WS composition:

- Prolog WS composition [16],
- SHOP2 system [12],
- OWLSXplan [18],
- WSPlan [14] and
- SEMCO-WS [23].

Among most frequently choice for AWSC belong utilization of OWL-S WS descriptions with planning task definition by PDDL language, and in next step solving this planning task by selected AI planner. Submitted work dealt with similar approach too, at which there is proposed utilization further OWL ontologies for planning state definition. These ontologies next with OWL-S WS descriptions present external specification of composition problem. Internal specification is represented by PDDL planning task. Own system proposal with transformation process description can be seen in next section.

5. System proposal for WS composition utilizing knowledge approaches

The proposal of own system for automatic web service composition come out from general proposal presented on fig. 1. Submitted proposal but includes specific standards, technologies and methods in regard to particular WS composition parts.

A proposal of own system will be marked by short cut KWSC - *K*nowledge oriented *W*eb *S*ervice *C*omposition. KWSC system (fig. 2) utilize knowledge approaches for WS composition problem definition. For WS descriptions were chosen in regard to AKWC analysis OWL-S ontological descriptions. From view of WS composition are given descriptions suitable mainly because they allow for inputs and output assign exact type definition from ontologies (e.g. if WS input is English word, in contrast to WSDL description, which allow define only fact that this word is string, OWL-S description allow utilize a concept from ontology, which is mapping to corresponding input from WSDL description). In additional to other possibility OWL-S descriptions allow define *precondition* (which must be fulfilled for WS execution) and *effect* (which will be changed after this WS execution). Besides this OWL-S WS description is in given proposal proposed utilization OWL ontologies for domain description. Each WS composition is realizing in some domain (e.g. word translator domain, travel domain, crisis management etc.). Therefore is convenient have described this domain by ontology and using this ontology for definition initial and goal state of the composition problem.

External specification defines composition problem by using *OWL ontologies* and *OWL-S WS descriptions*. Domain OWL ontologies and OWL-S descriptions of services are located in **knowledge data base**. By utilization domain OWL ontology user define *initial* and *goal state*, which are also represented as ontologies.

As *process generator* was chosen **AI planner**. There may be used random AI planner with condition that it is able to solve planning described by PDDL language. PDDL planning task represents internal specification of composition problem. In the system is then necessary transform external specification, which is described by semantic standards, into internal specification, which is described by PDDL planning task (see sec. 6). This transformation is performing in translator. **Translator** is tool, which transform OWL ontologies and OWL-S descriptions into PDDL planning domain and problem by available algorithm. Even though that a lot of authors dealt with WS composition by OWL-S and PDDL, there exists only little available information about how there is perform transformation and planning task creation (e.g. [12, 22]). The presented system additionally uses OWL ontologies for definition domain states. Therefore there was necessary introduction original algorithms, by which is able to transform composition problem described by using knowledge approaches into PDDL planning task. The selected AI planner in next step solves this generated planning task and provide plan, which consists from PDDL actions. PDDL actions were obtained from OWL-S processes. Therefore there is needed backward mapping obtained actions from plan into corresponding OWL-S processes and generate workflow. Likewise is necessary creating dataflow by mapping PDDL actions inputs and outputs from generated plan into corresponding OWL-S processes inputs and outputs. There is also possibility to transform this generated PDDL plan into OWL-S composed process, which consists from atomic processes or simpler composed processes.

Execution engine execute generated plan. Engine communicate with knowledge base, from which obtain OWL-S WS descriptions with their processes, and with **WS storage**. After plan execution the *result* is provided to user.

6. PDDL planning task creation from semantic problem composition description

For initial and goal state description are using OWL ontologies and for WS descriptions OWL-S descriptions. To-

[4] 1st International planning competition - 1998: ftp://ftp.cs.yale.edu/pub/mcdermott/aipscomp-results.html

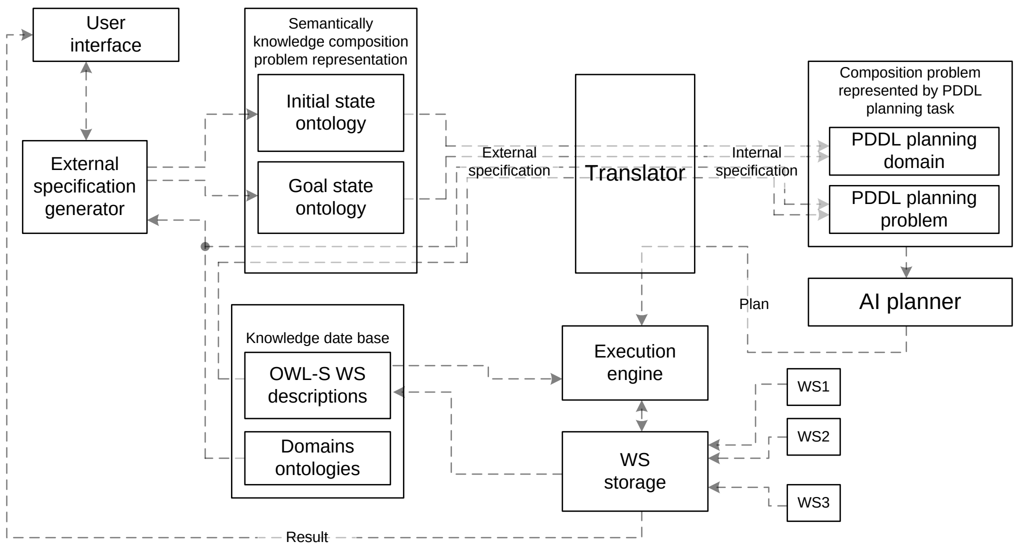

Figure 2: Own system proposal for AWSC

gether these ontologies (OWL and OWL-S) specify external specification of WS composition problems. Internal specification is specified by PDDL planning task. There are introduced transformation rules for transform external representation into internal. These rules are presented as algorithm, from which some were inspired by existing proposals for OWL-S description processing with planning task [17, 22]. PDDL structures will be described in EBNF[5] form. PDDL planning task consists from two parts [8]:

- planning domain and
- planning problem.

Each from this two parts will be handling particularly in next two subsections.

6.1 PDDL planning domain

Complete EBNF form for PDDL planning domain can be seen e.g. in [8]. For AWSC using knowledge approaches was EBNF form simplified. This simplified EBNF form of PDDL planning domain structure then looks likes is showing below:

```
<domain> ::= (define (domain <name>)
        [<require-def>]
        [<types-def>]:typing
        [predicates-def>]
        <action-def>*)
```

In the case of PDDL planning domain creation is necessary properly utilizing information, which is included in domain OWL ontology and in OWL-S WS descriptions. The PDDL domain creation problem may be divided into five parts. After these parts is for visualization presented this process on figure 3 and in this manner created example in example 4.

1. *Domain name* (`(domain <name>)`)

   This is the name of PDDL domain. This name may be obtained from domain OWL ontology header.

2. *Domain requirements* (`[<require-def>]`)

   This is an optional element of PDDL domain structure. In case that this element isn't include, is there automatically assumption one requirement, concrete STRIPS (`:strips`). In our case we create PDDL domain from OWL ontologies and therefore we need also other requirements, e.g. the possibility of type definition (`:typing`).

3. *Domain types* (`[<types-def>]:typing`)

   To PDDL types may correspond classes and a relationship class-superclass in OWL ontologies. PDDL types are created for each OWL class definition. In the case if OWL class has superior class (superclass), also the corresponding PDDL type has superior type. If superior class doesn't exist, superior type for corresponding PDDL type is object.

4. *Predicates* (`[predicates-def>]`)

   Predicate may represent property or relationship between entities. Predicates are in PDDL domain structures represented as atomic formula skeleton. Each this skeleton represents one predicate and consists from predicate name and a set of parameters. Predicates may be considered as patterns, by which is possible create facts. These facts are created by substitution using concrete object in predicate variable. Predicate will be created from domain OWL ontology. For PDDL predicates creation are using object properties, data properties and OWL classes definitions.

5. *Actions* (`<action-def>*`)

   Each PDDL action consists from name, parameter and action body. Action body next consists from

[5]Extended Backus-Naur Form - is using for a formal description computer programming languages and other formal languages

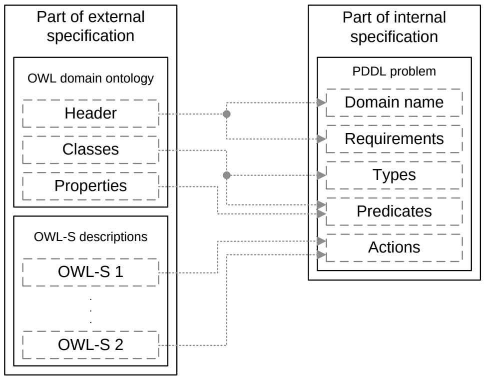

Figure 3: Creating PDDL planning domain from semantic described composition problem

precondition and effect. To create PDDL action will be used OWL-S WS description. The PDDL action name will be created from the WS operation name. Parameters will be defined by using input from OWL-S process description. OWL-S may contain three type of process:

(a) *Atomic process* is the simplest OWL-S process. From the view of user this process represent one step and there is a directly web service invocation. This WS is described by OWL-S.

(b) *Simple processes* are not directly executable. Likewise as atomic processes are simple processes considered as one step processes too. Simple process serves to abstract description of atomic and composite processes.

(c) *Composite process* is a process composite from subprocesses. A decomposition composite process into simpler parts is performing by given constructors (sequence, split, split-joint, choice, if-then-else, iterate, repeat until and repeat while). A composite process execution inhere in execution its parts, finally in execution atomic processes.

For each from these three types was proposed particular set of algorithm. Processes besides inputs and outputs may contain also preconditions and effects. These may be in OWL-S description represented by various definitions, e.g. most frequently by SWRL and KIF conditions. In own proposed system is assumption, that there are only SWRL conditions. By own created algorithms are retrieved from given SWRL conditions PDDL precondition and effects, which together create PDDL action body.

6.2 PDDL planning problem

Complete EBNF form for PDDL planning problem structure can be seen in [8]. As EBNF planning domain form also EBNF planning problem form was for AWSC using knowledge approaches simplified. This simplified EBNF form of PDDL planning problem structure then looks likes is showing below:

```
<problem> ::= (define (problem <name>)
        (:domain <name>)
        [<require-def>]
        [<object declaration>]
        [<init>]
        <goal>+)
```

```
(define
  (domain travel)
  (:requirements  :strips :typing)
  (:types road thing place at - object
         person vehicle - thing)
  (:predicates
    (at   ?has_place - place  ?has_thing - thing)
    (road   ?has_place_to ?has_place_from - place)
    (person   ?person - person)
    (vehicle   ?vehicle - vehicle)
    (thing   ?thing - thing)
    (place   ?place - place)
  )

  (:action Driving
    :parameters
      (?Thing - string
       ?FromPlace ?ToPlace - place)
    :precondition
      (and (road ?FromPlace ?ToPlace)
        (at ?FromPlace ?Thing))
    :effect
      (and (at ?ToPlace ?Thing)
        (not (at ?FromPlace ?Thing)))
  )
)
```

Figure 4: PDDL planning domain example

PDDL planning problem will be created from initial and goal OWL ontologies. The structure of PDDL problem consists from following definitions:

1. *Problem name* (`(problem <name>)`)

   A manner for obtaining PDDL problem name and requirements is similar as in case of PDDL domain creation.

2. *Domain name* (`(:domain <name>)`)

   Domain name represents the name of domain, which is in relation to PDDL problem.

3. *Problem requirements* (`<require-def>`)

   There are all requirement flags required in problem definition.

4. *Objects* (`<object declaration>`)

   Objects represent in PDDL problem concrete objects, which may be substituted on predicates parameters and create facts in planning task. By creating PDDL object are utilized classes instances (OWL individuals) from initial OWL ontologies.

5. *Initial state* (`<init>`)

   Initial state is in PDDL problem represented as a set of literals, which represent positive or negative atomic formulas. For PDDL predicates holds, that they are represented by atomic formula skeleton. By mapping concrete PDDL object to atomic formulas skeleton are creating atomic formulas, by which

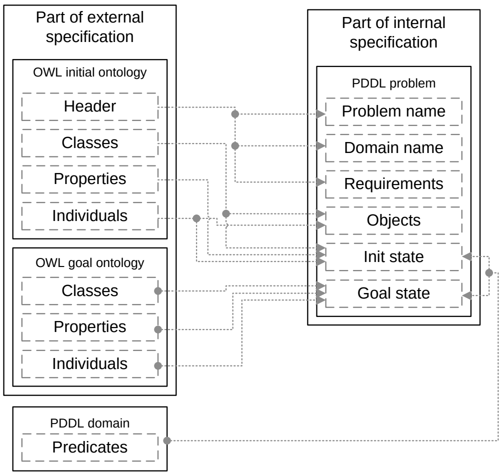

Figure 5: Creating PDDL problem from semantic described composition problem

there are represented facts in initial state. Therefore must be well-known predicates from PDDL domain.

6. *Goal state* (`<goal>+`)

   Goal state is in PDDL problem represented likewise as initial state by using atomic formulas. But additionally these may be in conjunction (**and**), disjunction (**or**) or negate (**not**) and combination of these operators.

On figure 5 is illustrated the process of PDDL planning problem creation from OWL ontologies. Simple example is next shown in figure 6.

```
(define
  (problem pb1)
  (:domain travel)
  (:requirements  :strips :typing)
  (:objects   a b c d - place  mazda - vehicle
              john - person)
  (:init
    (place a)
    (place b)
    (place c)
    (place d)
    (at b mazda)
    (at a john)
    (road a b)
    (road b a)
    (road b c)
    (road c b)
    (vehicle mazda)
    (road c d)
    (road d c)
    (person john)
  )
  (:goal
    (and (at d mazda)
      (at d john))
  )
)
```

Figure 6: PDDL planning problem example

7. Conclusions

Following objectives of this thesis is possible identify the following results:

1. In the work is presented a survey of WS together with semantic web standards and technologies, which related with WS and which are important for WS composition issue. There is also presents very general model for web services composition, which consists from several parts and one of mainly parts represents process generator. Generated processes are represented by workflow and dataflow. By execution this workflow is fulfilled WS composition problem.

2. There is described relation between composition problem and planning task definition. In the work is presented several main planning method suitable for WS composition and also description several existing proposal and system for (semi-)automated web service composition, which are compared. There was also presented one of most used planning languages, namely PDDL.

3. In practical part is showed own system proposal, which provide WS composition by utilization knowledge approaches as OWL ontologies and semantic OWL-S WS descriptions. Given system come out from analysis area of AWSC and existing systems for AWSC. For external composition problem specification are selected OWL ontologies. By using these ontologies are described domain states. Web services are described by using OWL-S ontologies. For internal composition problem specification was selected PDDL language. The main part of work focus on creation PDDL planning task from semantic ontologies (OWL and OWL-S). From external specification of WS composition problem, described by OWL and OWL-S, is needed generating PDDL planning task, which may be next solved by using AI planner. Therefore in work were introduced several algorithms, which perform creation of PDDL planning domain and problem.

4. Following theoretical proposal there are implemented selected algorithms. By using this algorithms may be presented suitability of proposed method for web service composition by using knowledge approaches.

7.1 Contributions

Major section in work dealt with transformation semantic described composition problem (by OWL and OWL-S ontologies) into planning task described by PDDL language. In the work was proposed utilization ontologically description of domain, in which is performing compositions. Further there is introduced several algorithms dealt with transformation problem from OWL and OWL-S description into PDDL planning task. OWL-S preconditions and effects definition are described by SWRL conditions. Given conditions are used to creation preconditions and effect of PDDL actions. PDDL planning task consists from domain and problem. Each from these two parts has several subparts. E.g. planning domain definition consists from types definition, predicates, actions and likewise. Planning problem contains objects, initial and goal state definitions. Among main contribution of this work belong processing WS composition by utilization knowledge approaches and proposal own system for WS composition with usage ontologically description of composition problem. In work is presented full transformation process of planning task creation, which may be solved at the end of this process by optional PDDL planner.

References

[1] H. Haas, A. Brown. Web Services Glossary. W3C Working Group Note, February 2004. http://www.w3.org/TR/ws-gloss/

[2] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana. Web Services Description Language (WSDL) 1.1. W3C Note. March 2001. http://www.w3.org/TR/wsdl

[3] M. Gudgin, M. Hadley, et.al. SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). W3C Recommendation, April 2007. http://www.w3.org/TR/soap12-part1/

[4] S. Bechhoefr, M. Dean, et.al. Owl Web Ontology Language Reference. W3C Proposed Recommendation. February 2004. http://www.w3.org/TR/owl-ref/

[5] D. Martin, M. Burstein, et.al. OWL-S: Semantic Markup for Web Services. W3C Member Submission, November 2004. http://www.w3.org/Submission/OWL-S/

[6] D. Martin, M. Burstein, O. Lassila, M. Paolucci, T. Payne, S. McIlraith. Describing Web Services using OWL-S and WSDL. http://www.daml.org/services/owl-s/1.1/owl-s-wsdl.html

[7] Nils J. Nilsson, Richard E. Fikes. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. Artificial Intelligence, 2(3):189-208, 1971.

[8] M. Ghallab, A. Howe, C. Knoblock, D.McDermott, A. Ram, M. Veloso, D. Weld, D. Wilkins. PDDL - The Planning Domain Definition Language, Version 1.2. Yale Center for Computational Vision and Control, Tech Report CVC TR-98-003/DCS TR-1165, October, 1998.

[9] Blai Bonet, Hector Geffer. Heuristic Search Planner Ver. 2.0. AI Magazine, Fall 2001, pages 77-80, 2001.

[10] Jorg Hoffman. FF: The Fast Forward Planning System, AI Magazine, Vol. 22-3, pages 57-62, 2001.

[11] Kutluhan Erol, James Hendler, Dana S. Nau. Semantics for Hierarchical Task-Network Planning. Technical report CS-TR-3239, UMIACS-TR-94-31, University of Maryland, March 1994.

[12] Dan Wu, Evren Sirin, James Hendler, Dana Nau, Bijan Parsia. Automatic Web services composition using SHOP2, In Workshop on Planning for Web Services, 2003.

[13] Joachim Peer. Web Service Composition as AI Planning - A Survey. Dissertation, University of St. Gallen, Switzerland, 2005.

[14] Joachim Peer. A PDDL Based Tool for Automatic Web Service Composition. Lecture Notes in Computer Science, Springer Berlin / Heidelberg. ISBN 978-3-540-22961-2, 149-163, september 2004.

[15] S. McIlraith, T. Son. Adapting Golog for Composition of Semantic Web Services. In Proceedings of the Eighth International Conference on Knowledge Representation and Reasoning (KR2002), 482-493, Toulouse, France, April 22-25, 2002.

[16] Evren Sirin, James Hendler, Parsia Bijan. Semi-automatic composition of web services using semantic descriptions. Web services: Modeling, Architecture and Infrastructure workshop in ICEIS 2003, Angers, France, April 2003.

[17] Evren Sirin, Parsia Bijan, Dan Wu, James Hendler. HTN planning for web service composition using SHOP2. Journal of Web Semantics 1 (4), pp.377-396, 2004.

[18] M. Klush, A. Gerber, M. Schmidt. Semantic Web Service Composition Planning with OWLS-Xplan. 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web, Arlington VA, USA, 2005.

[19] Andreas Gerber, Suleman Bull, Matthias Klush. OWLS-XPlan: OWL-S Semantic Web Service Composition Planner. User Manual Version 1.0, DFKI Saarbrücken, Germany: February 20, 2006.

[20] S. Oh, D. Lee, S. R. Kumara. A comparative illustration of AI planning-based web services composition. ACM SIGecom Exch. 5, 5 Jan., 2006.

[21] Evren Sirin. Combining description logic reasoning with AI planning for composition of Web services. Dissertation Thesis, University of Maryland, 239 p., 2006.

[22] Bo Yang, Zheng Qin. Composing Semantic Web Services with PDDL. Information Technology Journal 9 (1), ISSN 1812-5638, 48-54, 2010.

[23] Ondrej Habala, Marek Paralič, Viera Rozinajová, Peter Bartaloš. Semantically-Aided Data-Aware Service Workflow Composition. SOFSEM '09: Proceedings of the 35th Conference on Current Trends in Theory and Practice of Computer Science, ISBN 978-3-540-95890-1, p. 317-328, 2009.

[24] Resource Description Framework (RDF), http://www.w3.org/RDF/, 2004

[25] Avrim L. Blum, Merrick L. Furst. Fast planning through planning graph analysis. In Artificial Intelligence journal volume 90, 1636 -1642, 1997.

[26] Marian Babik. Optimizing description logic reasoning for the matchmaking and composition of semantic web services. Dissertation Thesis. Institute of Informatics, Slovak Academy of Sciences, Bratislava, p. 137, 2007.

[27] Malik Ghallab, Dana Nau, Paolo Traverso. Automated Planning Theory and Practice. Morgan Kaufmann Publisher Inc., San Francisco, CA, USA, ISBN 1-55860-856-7, 2004.

Selected Papers by the Author

Zoltán Ďurčík. Translation of semantic web services descriptions into a planning problem. In: *SCYR 2010 : 10th Scientific Conference of Young Researchers of Faculty of Electrical Engineering and Informatics Technical University of Košice* : proceedings from conference : May 19th, 2010, Košice, Slovakia. - Košice : FEI TU, 2010. - ISBN 978-80-553-0423-6. - S. 189-191.

Zoltán Ďurčík. Webové služby - kompozícia umelo inteligentnými metódami. In: *Znalosti 2010 : 9. ročník konference : Jindřichuv Hradec*, 3.-5. února 2010 : sborník příspěvku. - Jindřichuv Hradec : Vysoká škola ekonomická v Praze, 2010. - ISBN 978-80-245-1636-3. - P. 203 - 206.

Marian Babík, Martin Sarnovský, Zoltán Ďurčík. Establishing semantic annotation and execution of the text-mining services. In: *WIKT 2008 : 3rd Workshop on Intelligent and Knowledge Oriented Technologies* : zborník príspevkov. - Bratislava : STU, 2009. - ISBN 978-80-227-3027-3. - S. 78-82.

Zoltán Ďurčík. Kompozícia webových služieb - úvod do problematiky a návrh aplikácie. In: *WIKT 2009 : 4th Workshop on Intelligent and Knowledge oriented Technologies* : proceedings. - Košice : Equilibria, 2009. - ISBN 978-80-89284-42-9. - S. 15-21.

Marian Babík, Martin Sarnovský, Zoltán Ďurčík. Establishing semantic annotation and execution of the text-mining services. In: *GCCP 2008 : 4th international workshop on Grid Computing for Complex Problems* : October 27 - 29, 2008, Bratislava, Slovakia. - Bratislava : Ústav informatiky SAV, 2008. - ISBN 978-80-969202-9-7. - S. 104-112.

Zoltán Ďurčík. Automatic Web service composition. In: *SCYR 2009 : 9th Scientific Conference of Young Researchers : proceeding from conference.* - Košice : FEI TU, 2009. - 1 elektronický optický disk (CD-ROM). - ISBN 978-80-553-0178-5. - S. 122-125.

Zoltán Ďurčík. Automated web service composition. In:*WDA 2009 : workshop on data analysis : proceedings of the 9th International Student Workshop* , Čertovica, Slovakia July 2-4, 2009. - Košice : TU, 2009. - ISBN 978-80-89284-41-2. - S. 83-97.