A Goal-based Framework for Dynamic Service Discovery and Composition

Luiz Olavo Bonino da Silva Santos, Luís Ferreira Pires and Marten van Sinderen

Centre for Telematics and Information Technology, University of Twente
P.O. Box 217, 7500 AE Enschede, The Netherlands

Abstract. Service-oriented computing allows new applications to be developed by using and/or combining services offered by different organizations. Service composition can be applied when a client request cannot be satisfied by any individual service. In this case, the creation of a composite service from a number of available services could be pursued. This composite service should comply with the client's request in terms of functionality and expected results. This paper presents a goal-based framework for dynamic service discovery and composition. Our framework consists of a set of design principles and guidelines for service platforms to realize dynamic service discovery and composition.

1 Introduction

In recent years, goal-based analysis has been used in different areas of Computer Science to identify stakeholder's objectives, determine requirements for software systems and guide system's behavior. This generic approach ranges from techniques to accurately identify goals as presented in [1, 2] to the use of goals for service composition presented in [3, 4, 15].

Several different meanings can be found for the term goal, depending on the area of expertise being considered. Informally, one can define goal as a particular state of the world of which a stakeholder is committed to realize. From this informal definition, and considering the service-oriented computing principles, one can infer that goals can be used to represent service requests and therefore, they can be used to guide service discovery and composition activities. Initiatives applying goals for these purposes have been reported in [3, 4, 5]. These initiatives have in common the assumption that goal definitions are already available and have been previously identified and modeled. However, the initiatives do not clarify how these goals are gathered and modeled and how the goal descriptions relate to concrete services. We claim that prescriptions of how to model goals are still missing and that they should be addressed when developing a framework for dynamic service discovery and composition framework based on goal gathering and modeling.

This paper presents a framework for dynamic service discovery and composition based on the use of goals. The framework includes a precise definition of goal, techniques for goal assessment and modeling, and the rationale for dynamic service discovery and composition.

Olavo Bonino da Silva Santos L., Ferreira Pires L. and van Sinderen M. (2008).
A Goal-based Framework for Dynamic Service Discovery and Composition.
In *Proceedings of the 2nd International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing*, pages 67-78
DOI: 10.5220/0001899400670078

This document is further structured as follows. Section 2 presents and discusses definitions of the goal concept found in the literature, and proposes a definition that is appropriate for the service computing scenario. Section 3 presents approaches for service discovery and composition based on goals. Section 4 discusses our proposal for goal-based service discovery and composition, and Section 5 gives conclusions and identifies topics for future work.

2 Definition of Goal

The concept of goal has several different definitions varying from "*the result of scoring*" and "*the physical structure that defines where the score is achieved*" in some ball games to "*a statement of intent for the direction of the business*" in business administration. The definitions strongly depend on to which application area this term is applied. Narrowing down to the Computer Science domain, a variety of definitions of the goal concept can also be found. Below we introduce and discuss definitions of the goal concept found in the Computer Science literature, and present our definition of goal for the purpose of service discovery and composition.

2.1 Artificial Intelligence

In the Artificial Intelligence (AI) realm, goals are considered in problem solving research. The main motivation for this research is that in order to achieve their goals agents frequently need to act in the world. Considering computational agents, approaches and techniques have to be developed to support the generation of appropriate actions. The AI community has developed two complimentary approaches to generate these actions: planning and situated actions [9]. Planning is mostly used when a number of actions must be executed in a coherent way (pattern) to achieve a goal or when there are complex interferences of one action on others. Conversely, situated actions are used when the best possible action can be easily computed from the current state of the world, i.e., when no look ahead is necessary because actions do not interfere with each other. These approaches differ since planning uses a partial description of the state of the world while situated actions use the complete state space.

Both planning and situated actions consider a (complete or partial) description of the world, a goal description and a set of possible actions to change the world in the intended way. Therefore, goal is defined as a "*description of a world state that is expected to be realized*" [10]. More specifically, in planning a goal is defined as a partially specified state, represented by a conjunction of positive ground literals. A state s satisfies a goal g if s contains all the atoms in g (and possibly others as well). For example, the state $Rich \wedge Famous \wedge Miserable$ satisfies the goal $Rich \wedge Famous$.

2.2 Agent-Oriented Computing

In the agent-oriented computing community the term goal does not have a standard definition yet. In [11], a goal is defined as a "*state with highest utility and an agent*

must choose the course of actions to reach that goal". In [12], two types of goal models are presented, namely the task-oriented goal model and the state-oriented goal model. The former defines goal as a "*fixed list of tasks***"** and the goal satisfaction is achieved when the agent finishes all these tasks. The later defines goal as a "*final state that the agent tries to achieve by moving from its initial state through a defined and finite sequence of intermediary states*".

Another approach in the agent-oriented community that extensively uses goals is i* [13]. This approach defines two types of goal, namely hard goal and soft goal. A hard goal is defined as the "*representation of a stakeholder's intentional desire*". A soft goal is similar to a hard goal, except that it lacks clear-cut criteria for the goal's satisfaction, i.e., one cannot precisely determine whether a soft goal has been fully satisfied or not. The i* approach introduces mean-end diagrams in which tasks are the means for satisfying goals. In i*, goal satisfaction is not defined in the goal description, but it is indirectly described by a task related to that goal. However, it is unclear how goals should be formally specified in i*, since goals can appear in i* diagrams or textual descriptions. This lack of a formal definition leads to difficulties in using the approach consistently. Recently the consortium of universities and research centers involved in i* is working towards the definition of a common metamodel to tackle this issue.

Although these definitions of goal differ, a common agreement in the agent community is that goals are part of an agent's intentional properties and that the goal owner can act accordingly to satisfy the goal or can delegate the goal satisfaction to other agents. Agents commonly represent the stakeholders of a domain, being these stakeholders humans, organizations or software. Agents can delegate the fulfillment of a goal or the execution of a task. A goal delegation represents a situation where an agent has a goal, but for some reason it is not capable of fulfilling this goal, and delegates it to be fulfilled by another agent. Task delegation occurs similarly. The main difference between goal and task delegation is that while in the goal delegation the goal owner wants his goal fulfilled no matter how, in task delegation the task should be carried out in the way defined by the task owner.

Goal and task delegation can be used in goal-based service composition to guide the establishment of security and privacy constraints. For example, if a client application (the goal or task owner) defines that its goal or task can only be delegated to known agents, then the service platform should only select services from providers that have business relations with the client application's organization.

2.3 Service-Oriented Computing

The area of service-oriented computing also benefits from the use of goals, particularly to formulate users' requirements and to guide the service composition process. In [14], both these objectives are pursued by using a goal-driven approach to elicit functional requirements for inter-organizational processes and to identify which services should be provided by each organization. The approach uses a labeled directed graph with intentions as nodes and strategies as edges between intentions. Strategies are defined as ways to achieve intentions. An edge entering a node defines that the given strategy (the edge) can be used to achieve the corresponding intention (the node). Multiples edges entering a node represent alternative strategies that can be

used to achieve an intention. An intention is defined as a goal to be achieved by performing a process that consists of a sequence of intentions and strategies that should be followed to achieve a particular intention. Considering goals and intentions as equivalent concepts conflicts with other definitions that consider intention as part of the stakeholder's mental state towards a goal, in the sense that a stakeholder wants something to be achieved and has an intention to make it happen.

One major initiative in the Semantic Web service community is the Web Service Modeling Ontology (WSMO) [5]. WSMO is an ontology-based conceptual model for describing Semantic web services. In WSMO, a goal is defined as an objective that a client might have when consulting a web service, and describes aspects related to user desires with respect to the requested functionality. A goal in WSMO is described by non-functional properties, imported ontologies, used mediators, requested capabilities and requested interface. Since goals and web services descriptions have similar structures, service discovery and matching are facilitated. In the goal description, the capability section is used to express what the requester would like to achieve, and the interface describes how the requester would like to interact with a web service.

2.4 Our Definition

Considering the approaches to use goals for service composition discussed above, we can notice that there is no common agreement on the definition of goal. This definition is given at a high level of abstraction or in an informal way, often lacking precision and a formal representation. Another issue is that none of these approaches present a method or process to define goals and services. The approach presented in [14] shows a visual modeling notation, but the semantics and details of this notation are unclear.

In our approach, a precise definition of goal is needed to allow the specification of other elements, such as the metamodel and the domain ontologies, and to avoid ambiguity and misinterpretation when considering other definitions found in the literature. In this work, we define goal as a "*description of a stakeholder's intended state of affairs*", i.e., a stakeholder has a description of the world's setting that he wants to achieve and he is committed to act accordingly so that this setting becomes true.

In the domain of service-oriented computing, this definition means that a stakeholder (in this case a service client) has a description of the setting of the world he wants to see realized. Another important part of the goal definition is intentionality, which implies a commitment to procure the achievement of the goal. Again, in the case of the service-oriented computing, the achievement of the goal is delegated to services made available by third-parties, i.e., the service client will have its goal achieved by means of a concrete service provided by a service provider.

3 Goal-based Approaches

In Service-oriented computing literature we can find some initiatives for service discovery and composition based on goals. Among them, we consider the Web Service

Modeling Ontology (WSMO) [5], GoalMorph [6] and the approach presented in [15], which are further discussed below.

3.1 WSMO

The Web Service Modelling Ontology (WSMO) is a conceptual model that aims at describing the relevant aspects of Semantic web services [5]. The main objective of WSMO is to support the (total or partial) automation of the tasks normally related to web services integration, i.e., discovery, selection, composition, mediation, execution, monitoring, among others. WSMO is founded on the Web Service Modeling Framework (WSMF) [6] and refines and extends WSMF by means of a formal ontology and a family of languages.

WSMO is based on ontologies, as the well-accepted technology for knowledge representation. It also complies with Web technologies, in that it uses URI, namespaces, XML and other W3C recommendations. It supports decoupling between service requesters and service providers, offers mediation support to enforce interoperability, and explicitly separates service description from service implementation.

Following the WSMF guidelines, WSMO prescribes four main elements to describe semantic web services, namely ontologies, goals, web services and mediators. Ontologies provide (domain-specific) terminology for the other elements; goals define intentions of the service requester that should be solved by web services; web services represent atomic pieces of functionality that can be used to build more complex functionality; and mediators resolve interoperability problems between other components.

The WSMO conceptual model is supported by a family of languages called the Web Service Modelling Language (WSML) [7]. This family of languages consists of WSML-Core, WSML-DL, WSML-Flight, WSML-Rule and WSML-Full, which offer different levels of expressiveness, varying from minimal level supported by WSDL-Core (intersection of Description Logics and Logic Programs, [8]) to the maximum level expected to be supported by the (yet to be released) WSDL-Full (First-Order Logic combined with Logic Programs).

WSMO distinguishes the concepts of web service and service. A web service is a computational entity that is able to achieve the goal of a requester, while a service is the actual value provided by the business service provider [5]. In WSMO, a web service description defines the capabilities and the interface of a web service.

In WSMO, a goal defines what the requester expects to achieve by invoking a web service. Like the web service description, a goal description contains information about the capabilities and the interface, but in this case the capabilities are used to describe what the requester would like to achieve, and the interface is used to describe how the requester would like to interact with a web service. Goals and web services descriptions have similar structures, facilitating in this way service discovery and matching. In WSMO, goal descriptions can be viewed as the counterparts of web service descriptions, in the sense that a goal description states what the service requester wants to get achieved, and a web service description specifies what a web service can achieve.

3.2 GoalMorph

GoalMorph [4] is a context-aware goal transformation framework. The framework uses planning techniques and combines goals and contextual information to derive service compositions. Service composition is achieved in GoalMorph by using goals to represent information about the planning problem (or composition request) and to limit the inference space in the planning process. The reasoning on which goals states should be satisfied provides criteria for creating a successful plan.

The main contributions of GoalMorph are (*i*) a model that represents context-aware goals; (*ii*) analysis and taxonomy of goal types and corresponding transformations; (*iii*) a model that allows reasoning about partial satisfaction of goals; and, (*iv*) a utility model to reason about goal transformations and corresponding partial success in achieving one goal against partial success in achieving another goal.

GoalMorph classifies goal conditions into *core goals*, which arise from a user's request, and *context goals*, which are side-effects of the current user's context. Moreover, a taxonomy of goals is presented and consists of:

- *Core Goal*: a goal condition that purely describes the user's task intention.
- *Base Core Goal*: the absolute minimal core goal condition that needs to be satisfied to achieve a viable solution. Other core goals that are not base core goals can be suppressed from a plan to achieve a partial goal satisfaction.
- *Dependent Context Goal*: a context goal condition which works as the attribute of a core goal condition. If the core goal is removed from the goal set, the related dependent context goals are also removed.
- *Independent Context Goal*: a context goal condition that does not directly affect the user's request and, therefore, is not removed in case a core goal is removed.

The starting point of GoalMorph is the user's request for a composite service. A user selects a goal from the GoalRepository and this goal forms the basis for a composite service. The GoalRepository stores the available core goal templates that have been defined by domain engineers using the planning language. GoalMorph uses the Planning Domain Description Language (PDDL) [16] as a planning language to represent goals. ContextService is a general middleware infrastructure for context brokering from a number of different context sources to context consumers. The context source used in GoalMorph to retrieve contextual information is based on the solution proposed in [17]. The ContextProxy component generates context goal conditions that constrain the composition request based on contextual information of the user provided by the ContextService.

Based on the core goal conditions from GoalRepository and the context goal conditions from ContextProxy the final composition request is assembled. Then, the Planner Composition Engine receives the problem definition (the final composition request) and the domain definition from the ServiceRegistry, and uses knowledge about available actions and their consequences to identify a solution, i.e., to create a plan. The creation of the plan fails if the goal cannot be satisfied or if the domain knowledge is incomplete.

If the Planner Composition Engine fails to create a plan for the problem definition, control is transferred to the Goal Transformation Engine, which transforms the goal into a problem solvable by the Planner Composition Engine. The transformation is performed on the core goal(s) and, by interacting with the GoalRepository, the Goal

Transformation Engine attempts to find forms of the original goal that can be satisfied. An ontology consisting of a number of hierarchies for the goals stored in the GoalRepository is used to support goal transformation.

When a transformation ends successfully, the transformed goal is passed to the ContextMesh for context layering, i.e., for refining the context goal conditions by context unfolding or relaxing the context goal conditions by context folding. Users provide context importance measures through a GUI, and these measures are used as guidelines for the context layering operations. The final step is to feed the transformed goal to the Planner Composition Engine, and the resulting composition is evaluated by the user to refine the goal transformation utilities.

3.3 A Goal and Task-based Approach for Service Composition

Another approach for goal-based service composition is presented in [15]. This approach starts by building a task-oriented semantic representation model of web services. In this model, application scenarios and task goals are defined at a higher abstraction level. Dynamic service discovery and matching, and goal-based composition are performed to achieve the user goals. A Task Definition Language (TDL) has been specified. In TDL tasks can be decomposed into sub-tasks and tasks are defined to achieve goals. An ontology is used to provide an uniform vocabulary for the description of concepts and tasks. A goal consists of activities. Each activity can be defined more specifically. A basic activity can be matched directly with a concrete service and does not need to be further specified.

The approach divides the service composition life-cycle into three phases: definition, construction and execution. The definition phase allows defining goals and abstract tasks to achieve the goals. The difference between the abstract definition and concrete service composition is that service providers and control information are not specified in the abstract definition. In the construction phase, the task is decomposed, concrete services are matched and bound, and an executable service process is generated. In the execution phase, the concrete web services are invoked and the process is executed to achieve the user's goal. In [15] it is unclear how a goal should be defined, and the terms goal and task are used interchangeably.

4 The Proposed Framework

Our approach to dynamic service discovery and composition is based on goal modeling, and assumes that the involved stakeholders (client applications, service providers, third-party entities) are placed in a common and known domain. This requirement is necessary because the approach relies on the availability of domain-specific ontologies. For each domain, valid goals for the stakeholders of that domain are identified together with the tasks required for their fulfillment. Fig. 1 depicts the following main elements that comprise our framework:

– *Service Platform*. A service platform supports the interaction between service providers and service clients. From the service provider's perspective, the platform supports the publication of service descriptions. From the service client's perspec-

tive, the platform provides facilities for service discovery, composition, invocation and monitoring, among others. Different platform implementations can offer different sets of facilities. In this work we assume that the platform offers at least service discovery and composition.

- *Goal-based Service Ontology*. This ontology defines domain-independent concepts such as service, stakeholder, organization, goal and task, and their relations. These definitions are further used and specialized in the domain and task ontologies.
- *Domain Ontologies*. These ontologies define domain-specific concepts and the relations among these concepts. Domain ontologies should be available for the service platform as well as for the service providers and clients.
- *Task Ontologies*. Associated with domain ontologies, task ontologies provide domain-specific definitions of the tasks valid in a domain.

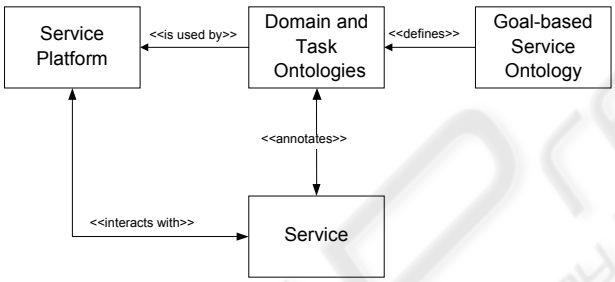

Fig. 1. Main components of our approach and their relationships.

4.1 Service Discovery

Our approach defines a set of steps that have to be followed in order to provide support for service discovery. Assuming the existence of the goal-based service ontology and the service platform, the necessary steps to be taken and the stakeholder responsible for each step are the following:

1. *Domain Specification*: A domain expert defines domain and task ontologies with the correspondent concepts, relations and the valid goals and tasks of the domain. We illustrate this with the meeting organizing domain. In this domain, concepts like attendee, organizer and meeting should be defined. The relations among these concepts should be defined as well, such as the attendance relation between an attendee and a meeting or the organization relation between a meeting and its organizer. An example of a valid goal in this domain is to have a meeting scheduled given a desired list of attendees and tentative time slots. Lastly, tasks pertinent to this domain are modeled, such as schedule a meeting, accept a meeting invitation, reject a meeting invitation and reschedule a meeting.
2. *Service Description*: Service providers describe their services supplying information about the behavior of the service, the expected inputs and outputs as well as the pre and post conditions for the service. The service description is annotated with the concepts defined in the domain ontology. Moreover, a mapping is established between the offered service and a task defined in the task ontology. In this

way, the platform can identify which available services implement the behavior of which tasks.

3. *Goal or Task Request*: The service clients can request some service to the platform in two alternative ways: by defining a goal and requesting the platform to return a service that fulfills this goal, or by defining a task and submitting it to the platform.

In case a goal is defined and submitted to the platform, the platform performs the following procedure:

a. It tries to match the user goal with the goals defined in the domain ontology;
b. Once a goal is found, the platform tries to find tasks defined in the task ontology that can fulfill this goal;
c. With the mapping between services and tasks established in step 2 above, the platform can identify an available service that implements the task that fulfills the requested goal.

In case a task is defined and submitted to the platform, the platform first tries to find a matching task in the task ontology and then services that implement the behavior of the task.

Fig. 2 presents a sequence diagram depicting the interactions between the service client and the service platform for service discovery. The diagram only shows the interactions up to the point where a service is found. If the platform plays a broker role [18], the platform also performs service invocation, service composition and adjustments of the results (if needed), and finally returns to the client the final service execution results.

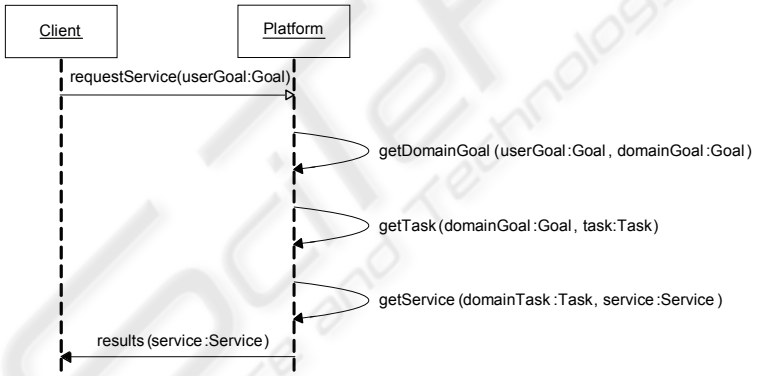

Fig. 2. Interaction between the client and the platform for service request.

4.2 Service Composition

In case the platform cannot find a task to fulfill a given goal or a service realizing a domain task, further steps should be taken. Depending on where the mismatch happened we have to perform task composition (when a task is not found) or service composition (when a service is not found). In both cases the steps are similar and rely on the granularity of the goals and tasks specification. In case a task is not found to fulfill a given goal, the platform checks the goal decomposition specification and tries to find tasks that fulfill the sub-goals.

Fig. 3 depicts an example of goal and task decomposition. The figure uses the i* notation [12], in which circles represent stakeholders, ovals represent goals, hexagons represent tasks and dashed circles representing stakeholder's boundaries. The links between goals and sub-goals, and tasks and sub-tasks represent goal and task decomposition, respectively. The links between goal and task represent fulfillment. Fig. 3(A) depicts the goal modeling of a meeting organizer who has the goal of organizing a meeting. This goal is further decomposed into *invite attendees* and *book venue.* Fig. 3(B) depicts the *invite attendees* sub-goal, which is fulfilled by as *invite attendees* task. Although in this example the sub-goal and the task have identical names, they represent different concepts. The sub-goal *invite attendees* represents a setting of the world where the attendees of the meeting are invited, while the task *invite attendees* represents a process that should be followed to invite the attendees for the meeting. In Fig. 3, the *invite attendees* task is further decomposed into three sub-tasks namely *get attendees' list, send invitation* and *confirm attendee attendance.* With these two models, the platform is able to perform either goal (or sub-goal) matching or task (or sub-task) matching.

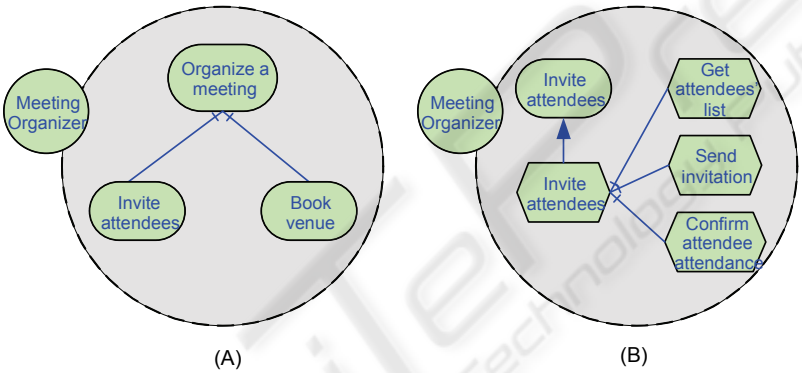

Fig. 3. Goal and task decomposition.

5 Conclusions and Future Work

In this paper we present a goal-based approach for dynamic service discovery and composition. The approach is based on a behavior model represented by goal modeling. Goals can be further decomposed into sub-goals, and tasks fulfill (sub-)goals. Client's service requests are represented in terms of goal specifications and are submitted to the supporting service platform. Service discovery is performed by the platform by matching the submitted goal against existent domain goals defined in a domain ontology. Another possibility is that the service client does not submit a goal but a specific task. In this case, the platform performs service discovery by matching the user-specified task against existing tasks defined in a task ontology.

This approach is founded in a well-defined set of domain and task ontologies. The domain ontology describes the concepts and relations valid for a specific domain, while the task ontology defines valid tasks for that domain. These ontologies are also defined in terms of a higher-level goal-based ontology, which specifies concepts such

as stakeholders, goals, tasks and their relationships, and can be used to created domain and task ontologies of different domains.

This paper presents the main components of the approach, namely a goal-based service ontology, domain and task ontologies and the service platform, and the steps necessary to achieve dynamic service discovery and composition. The next steps of this ongoing research are: (*i)* defining a goal-based service ontology; (*ii*) defining domain and task ontologies for a chosen domain; and, (*iii*) implementing the service discovery and composition mechanisms of the service platform in order to evaluateand validate of the approach.

Acknowledgements

This work is partly funded by the Freeband Communication project A-MUSE (http://a-muse.freeband.nl). A-MUSE is sponsored by the Dutch government under contract BSIK 03025. The work on the definition of the goal concept is being conducted in cooperation with Dr. Giancarlo Guizzardi, Universidade Federal do Espírito Santo, Brazil.

References

1. Bresciani, P., Perini, A., Giorgini, P., Giunchilia, F., Mylopoulos, J., Tropos: An Agent-Oriented Software Development Methodology. Journal of Autonomous Agents and Multi-Agent Systems, vol. 8, pp. 203-236, The Netherlands, 2004.
2. Antón, A. I., Goal-Based Requirements Analysis, 2nd IEEE International Conference on Requirements Engineering (ICRE 96), Colorado Springs, Colorado, USA, pp. 136-144, 15-18 April 1996.
3. Padgham, L., Liu, W., Internet Collaboration and Service Composition as a Loose Form of Teamwork, Journal of Network and Computer Applications, vol. 30, issue 3, pp. 1116-1135, Academic Press Ltd., London, UK, August 2007.
4. Vukovic, M., Robinson, P., GoalMorph: Partial Goal Satisfaction for Flexible Service Composition, IEEE International Conference on Next Generation Web Services Practices, Seoul, Korea, August 2005.
5. Feier, C. (ed.) Web Service Modeling Ontology Primer. W3C Member Submission. 3 June 2005. Available at http://www.w3.org/Submission/2005/SUBM-WSMO-primer-20050603/.
6. Fensel, D., Bussler, C. The Web Service Modeling Framework WSMF. White Paper, http://www1-c703.uibk.ac.at/users/c70385/wese/wsmf.paper.pdf, last accessed November 8, 2007.
7. de Bruijn, J. and Lausen, H. (eds.) Web Service Modeling Language (WSML). W3C Member Submission. 3 June 2005. Available at http://www.w3.org/Submission/2005/SUBM-WSML-20050603/
8. Grosof, B.N., Horrocks, I., Volz, R. and Decker S. Description logic programs: Combining logic programs with description logic. In Proceedings of the Twelfth International World Wide Web Conference (WWW 2003). ACM, 2003, pp 48-57.
9. Weld, D. S., An Introduction to Least Commitment Planning, The AI Magazine, nº 4, vol. 15, pp. 27-61, Winter 1994.
10. Russel, S., Norvig, P., Artificial Intelligence: A Modern Approach, Prentice Hall, 2nd edition, December 2002.

11. Moghadasi, M. N., Haghighat, A. T., Ghidary, S. S., Evaluating Markov Decision Process As A Model For Decision Making Under Uncertainty Environment, in Proceedings of the 2007 International Conference on Machine Learning and Cybernetics, vol. 5, pp. 2446-2450, 19-22 Aug 2007, Hong Kong, China.
12. Rosenschein, J. S., Zlotkin, G., Rules of Encounter: Designing Conventions for Automated Negotiation among Computers, MIT Press, Cambridge, Massachusetts, USA, 1994.
13. Yu, E., Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering, In Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE'97), pp. 226-235, Washington D.C., USA, Jan 6-8, 1997.
14. Kaabi, R. S., Souveyet, C., Rolland, C., Eliciting service composition in a goal driven manner. In Proceedings of the 2nd International Conference on Service-Oriented Computing (ICSOC '04). ACM Press, New York, NY, USA, pp. 308-315, 2004.
15. Zhang, K., Li, Q., Sui, Q., A Goal-driven Approach of Service Composition for Pervasive Computing., in Proceedings of the 1st International Symposium on Pervasive Computing and Applications, pp. 593-598, August 3-5, 2006.
16. Ghallab, M., Howe, A., Knoblock, C., McDermott, D., Ram, A., Veloso, M., Weld, D., Wilkins D., PDDL – The Planning Domain Definition Language, (1998).
17. Lei, H., Sow, D.M., Davis, II, J.S., Banavar, G., Ebling, M.R.: The design and applications of a context service. SIGMOBILE Mob. Comput. Commun. Rev. 6 (2002) 45–55.
18. Bonino da Silva Santos, L.O., van Sinderen, M., Ferreira Pires, L., Architectural Models for Client Interaction on Service-Oriented Platforms, in Proceedings of the 1st International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing (ACT4SOC 2007) in conjunction with the 2nd International Conference on Software and Data Technologies (ICSOFT 2007), Barcelona, Spain, July 22, 2007.