Decision Making in Non-Deterministic Environments

George Markou
Dept of Applied Informatics
University of Macedonia
Thessaloniki, Greece
gmarkou@uom.gr

Abstract

My PhD research is focused on Web Service Composition through the use of AI planning techniques, considering the domain as an inherently non-deterministic one. The main goal of my research is to build a system in which an end user will be able to search for suitable web services, add new ones, and create new composite ones either manually or automatically. As a practical example of the usefulness of such a system we plan on integrating it into a personal time management platform; the goal of this line of research is to allow for a user to indicate a task that he wants to perform on his personal time and the system being able to produce composite web services that may achieve such a task automatically. The results of my work so far include the implementation of an online platform that has an integrated web service registry and allows its users to semi-automatically create a new, composite web service. We are currently working on non-deterministic algorithms that will be integrated into the automatic web service composition module.

Introduction

The use of Web Services (WSs) has provided a way to develop flexible and robust systems with reduced costs and requiring less time than building new, complex applications from scratch. However, WSs exist and operate in an ever-changing and expanding environment, and as such it is difficult to expect from a human user, or even an expert, to manually complete the goal of a Web Service Composition (WSC) process.

The number of WSs is growing continuously and, as such, the web services' discovery phase becomes more difficult. Web services can change interfaces or even part of their usage multiple times throughout their lifespan; even if they remain static, there is always the possibility that their execution is not successful. A WSC process should automatically detect and respond to such changes in a way that a human will probably not be able to.

Our line of work has focused on implementing an efficient WSC process that takes into account the domain's non-determinism, and making use of such a process in a personal time management system. In the case of WSC, the most widespread approach to solving the problem is to compile it into an AI planning one and use the already available tools and techniques of that field of research. The reason behind the popularity of such approaches is the fact that the AI planning community is more advanced than that of the WSC community, and has produced results that have become extremely more efficient and scalable in the past few years.

So far, an online platform has been implemented that allows its users to create workflows that can be converted into composite web services; moreover, a detailed evaluation process has been defined. We are currently working on a non-deterministic algorithm that will be used for the automatic web service composition process. In the future, this algorithm will be integrated in the existing application and evaluated against the manual module. The rest of my research will be focused on using the aforementioned system in combination with a personal time management one, so that some of the tasks inserted in it can be achieved automatically through composited web services.

Past Work

Our work so far has been towards implementing a WSC system that would exhibit the following functionalities:

- Advertisement of a new WS in a registry, as well as online editing and retrieval of the WSs already stored.
- Generation of a workflow based on OWL-S' (Martin et al. 2004) control constructs and the WSs stored in the online registry.
- Translation between the language used to store the output workflow internally and the one used to describe the semantic WSs taking part in the composition, namely BPMN 2.0 (Object Management Group 2011) and OWL-S, respectively.

- Evaluation of the output WSC process, based on quantitative criteria (e.g., the number of WSs considered for the composition, the transformation time of the WSC domain to a planning one, or the total planning time) and pre-defined use case scenarios.

In accordance with (Sirbu and Hoffmann 2008) we consider WSC at the functional level, i.e., only taking into account OWL-S' service profiles. In summary, the inputs and outputs of WS are mapped to input and output parameters on which preconditions and effects are specified. A more detailed analysis on the translation process between the web service and the AI planning domain is presented in the next section.

Online Registry

Web services' technologies are based on the idea of maximizing the reuse of loosely coupled components. As such, our view is that the systems implementing web services' functionalities should also be created with the same approach in mind and incorporate already freely available components as their sub-elements. Apart from the additional effort required to create a new component from scratch, such approaches have led to an abundance of applications and standards that only slightly differ from each other, while making the quantitative comparison of different systems difficult; this fact is illustrated by various surveys relying only on qualitative criteria to review the available methodologies (Chan, Bishop, and Baresi 2007).

Our system's implementation is based on such a notion, and as such, insures maximum conformance to the current WS standards and facilitates its quantitative evaluation and comparison to other WSC systems. To our knowledge, no other open source web-based WSC system exists.

The existing system supports various functionalities relating to different stages of WSC; the first one is the ability to store the service descriptions that will be used later in the discovery of suitable WSs in a registry. We opted to use iServe (Pedrinaci et al. 2010) as the core of our application and not a UDDI (Clement et al. 2004) approach, since its search functionality does not support the semantic content of Web services. iServe is an open platform for publishing and discovering services that supports importing service annotations in a range of formalisms (e.g., SAWSDL, WSMO-Lite and OWL-S) through their transformation to linked data expressed in terms of a simple, common, vocabulary for services.

Specifically, we make use of its web-based application that allows users to browse, query and upload services, which, in our case, are semantically described in OWL-S. We have added an online XML editor to the application, made several improvements to its interface and functionality, and populated the registry with version 4.0 of the OWL-S Service Retrieval Test Collection (OWL-S TC) (SemWebCentral 2010). Fig. 1 illustrates the basic functionalities of the implemented registry and its sub-components.

Evaluation

As mentioned in the previous section, the recent bibliography suggests a gap in the evaluation process of the current WSC systems (Chan, Bishop, and Baresi 2007). Not only is there no standard web service test set (Hoffmann et al. 2009), but most approaches, especially the ones related to planning based techniques, simply evaluate their methodology on a single case study, without

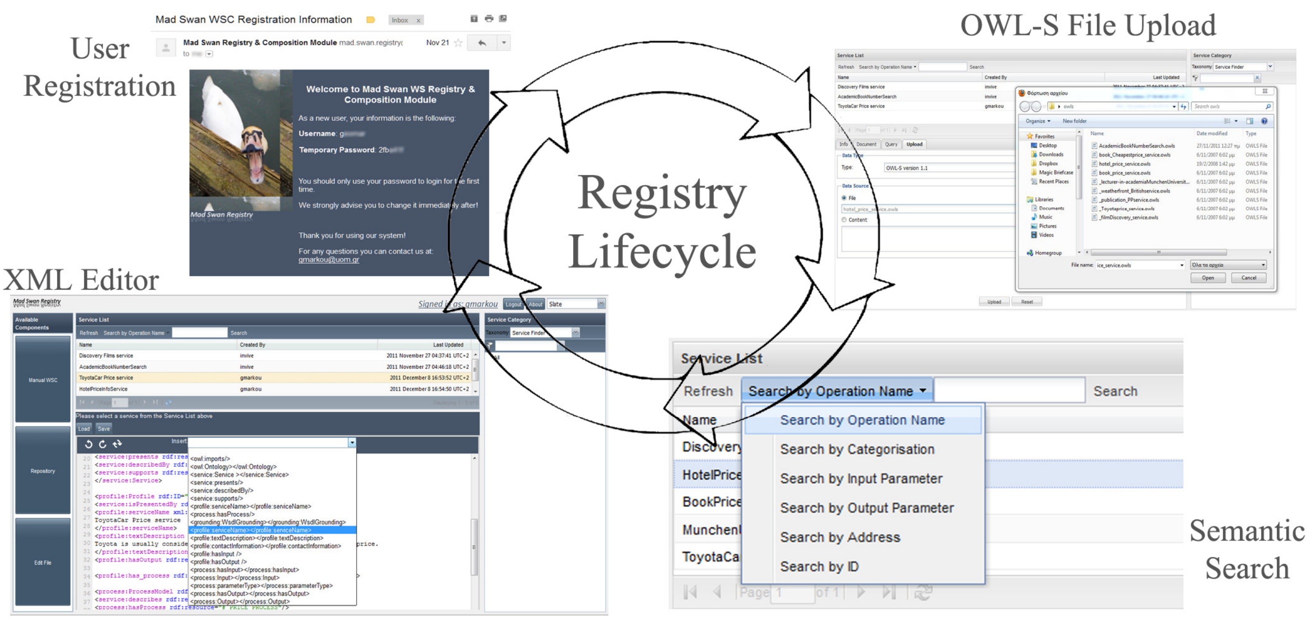

Figure 1. Registry Lifecycle – Implemented system screenshots.

referring to quantitative criteria (McDermott 2002; McIlraith and Son 2002). Only recently, a few approaches, such as (Hoffmann et al. 2009; Kona et al. 2008), have deviated from the rule of evaluating their methodology without quantitative criteria.

Use Case Scenarios

The lack of a standard web service test bed, concerning both the scenarios used to test the WSC process, and the web services that take part in it is detrimental to the WSC systems' development, as it is currently not possible to evaluate a WSC approach efficiently and objectively against another one. Moreover, this fact has led to the trend of most WSC approaches not providing any quantitative data at all concerning their system's performance. However, in the past few years OWL-S TC has been used extensively, as a test set in the recent S3 contests (S3 Contest 2011), or in several aproaches in the recent literature (Kuzu and Cicekli 2012; Mesmoudi, Mrissa, and Hacid 2011), a fact that suggests its suitability for use in our evaluation experiments.

We believe that the definition of specific use case scenarios in detail, as well as the provision of the actual web services' descriptions that will be used, benefit both the efficiency of our own approach as well as the WSC community, as it will allow for the reproducibility of our experiments and the comparison of existing WSC systems with each other. As such, we have designed three use case scenarios, each based on the web services contained in a single domain of OWL-S TC, and with an increasing amount of non-determinism and complexity than the previous one.

In order to design useful test cases for our system, we made several minor modifications to the available web services' descriptions and their relative ontologies, and also added a few descriptions to the collection, albeit similar to the ones already included in it. A full description of the use cases and the web services they are based on can be found in (Markou 2012).

The first use case is fully deterministic, allowing for the output of a fully serialized composite web service; it refers to a user who knows part of a movie title and wants to retrieve all the comedy films that exist with a similar title, along with their pricing information. This use case uses the web services in the "Communication" domain of the test collection, with the relevant ones in regard to its scenario amounting to a total of 58 semantic web services.

The other two scenarios feature non-deterministic elements, such as preferences between types of products, or cases where a web service may have different outcomes. Particularly, the second one refers to an online bookstore user who wants to purchase a book with a specific method of payment (with three different choices being available), with the output composite web service having different outcomes depending on whether the book is in stock at the store or not.

If the book is available, the composite web service should add it to the user's shopping cart, purchase it with the specified method of payment, and output information regarding it, such as its size and if a reader of the book has written a review regarding it. If, however, it is not in stock, no payment should be made, and no further information concerning it should be displayed to the client. This use case uses the web services in the "Education" domain of OWL-S TC, with the relevant ones being 285 in total.

The final use case also concerns the purchase of an item, but in this case more than one sellers are assumed to exist, and the composite web service should check with all of them to determine if the item is in stock. Moreover, this scenario differs from the previous one in that is also takes into account the user's preferences. In specific, the user is assumed to have a preference towards an analog SLR camera model, but is willing to settle for other cameras if that specific one is not in stock.

As such, if a store is found that sells the analog SLR model and has it in stock, that product should be added to the user's shopping cart. If it is not in stock, the search should continue for another store that sells it, and if one cannot be found, the process should be repeated, this time searching for the camera's compact version, or, if all else fails, for any camera available in stock. The third use case scenario makes use of the test collection's "Economy" domain and of a total of 359 semantic web services.

Although the first two scenarios can be considered as special cases of the last one, it is important to showcase that the system can indeed cope with the generation of both sequential and conditional plans, with and without preferences. Moreover, the importance of the scenarios lies in that they exhibit that this particular set of web services can be used to produce meaningful use cases that can evaluate the capabilities of WSC approaches efficiently and in a manner that is reproducible and extensible.

Manual Web Service Composition Module

Since the literature does not suggest a standard test bed or any WSC systems that are directly comparable to ours in terms of motivation (that is, regarding the WSC problem as an inherently non-deterministic one), use of standards and test sets/use case scenarios, we decided to further evaluate our automatic WSC approach against a manual one.

Specifically, we made use of an existing open source BPMN 2.0 modeler, Petals BPM (EBM Websourcing 2012), which we modified so as to satisfy the needs of a manual OWL-S composer; that is, we created the necessary OWL-S constructs, as well as some "helper" constructs that are used to provide a more intuitive interface.

The OWL-S constructs currently supported by the application are the ⟨*Sequence*⟩ (implicitly), ⟨*If-Then-Else*⟩, ⟨*Split+Join*⟩, and ⟨*Repeat-While*⟩ control constructs, along with the necessary inputs, outputs and web services' elements. The "helper" constructs comprise of an

⟨*End Split+Join*⟩ and an ⟨*End Repeat-While*⟩ construct used in conjunction with the regular ⟨*Split+Join*⟩ and ⟨*Repeat-While*⟩ constructs to enclose other elements in them, and dedicated ⟨*If*⟩ and ⟨*Else*⟩ sequence flows that are only used along with an ⟨*If-Then-Else*⟩ gateway. Moreover, there are ⟨*Start*⟩ and ⟨*End*⟩ constructs to signify the beginning and end of a workflow. That is, with the exception of input data that can precede a ⟨*Start*⟩ construct to signify that the input is valid for the whole workflow, no other construct can be used before a ⟨*Start*⟩ construct or after an ⟨*End*⟩ one.

Since the purpose of the automatic WSC module is to help even a non-expert user create composite web services, our aim was to implement the manual WSC module with the same principal in mind. For this reason, during the manual creation of the workflows the users are guided in regard to the correct use of the available constructs whenever they input one in a workflow. Moreover, the created graphical workflows are also validated against pre-defined rules whenever the users save them.

Some of the rules, such as that a ⟨*Start*⟩ construct must have at least one outgoing sequence flow and the ⟨*End*⟩ construct must have at least 1 incoming sequence flow, were maintained from the original Petals BPM application. Others, such as that an ⟨*If-Then-Else*⟩ construct can only have one outgoing ⟨*If*⟩ sequence flow and one (optional) ⟨*Else*⟩ sequence flow, or that every ⟨*Split+Join*⟩ construct must be accompanied by the related ⟨*End Split+Join*⟩ construct, were added in order to help the user export a valid composite web service.

Finally, the web service constructs that are added to the workflows can be bound to specific web services already in the registry of our online application, and in a similar manner, the data input/output constructs that are used can also be bound to relevant ontologies' concepts present in it. Fig. 2 illustrates part of the interface of the manual web service composition module along with the workflow representing the second use case scenario.

Current Research

Since PDDL and OWL-S are, respectively, the de facto planning language and the most widely used semantic description language, it is to be expected that several attempts that utilize them together in order to automatically solve WSC problems exist. Moreover, as the latter has been heavily influenced from planning languages, a (perhaps partial) mapping from OWL-S to PDDL is relatively intuitive.

We plan on using PPDDL (Younes and Littman 2004), the planning language used in the non-deterministic tracks of the 2006 and 2008 International Planning Competitions for the purposes of the automatic WSC. PPDDL is essentially a syntactic extension of PDDL2.1, capable of modeling non-deterministic actions through the introduction of probabilistic effects, which can be arbitrarily interleaved with conditional effects and universal quantification. For the time being we have created two planning domains, based on variations of the online bookstore scenario, each of which is available in both (PPDDL) probabilistic and deterministic (PDDL) versions (Markou 2013).

Since the WSs in the registry are described semantically through OWL-S, we have to adopt a translation pattern between the two languages. OWLS-Xplan (Klusch, Gerber and Schmidt 2005) is arguably the most known approach incorporating such a translation. The system presented includes a conversion tool that translates OWL-S descriptions to corresponding PDDL 2.1 ones; we plan to follow an approach similar to that one and analogous to that in (Hatzi et al. 2011).

A second approach is presented in (Kuzu and Cicekli 2012); its authors acknowledge that the WSC process cannot ignore the inherent non-determinism in the domain, and present a methodology that interleaves planning and execution to tackle it, based on an existing PDDL planner (Simplanner).

After the aforementioned translation, AI planning techniques can be used to generate the output plan/composite web service. Since composite web services

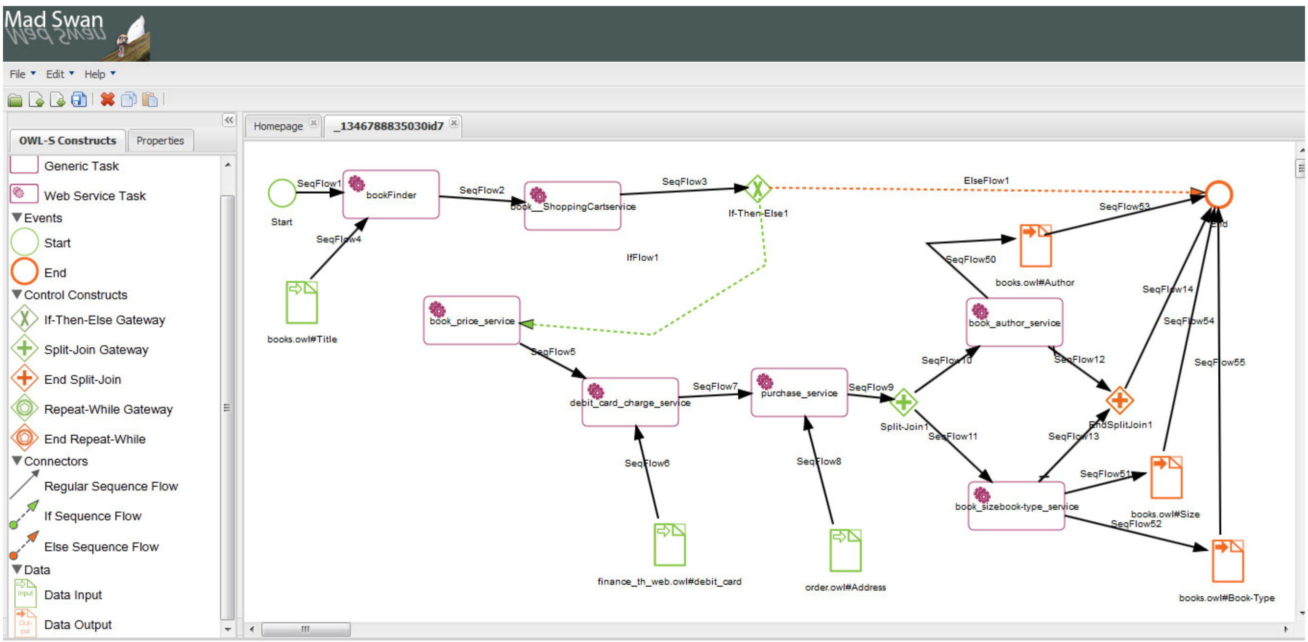

Figure 2. Online bookstore scenario workflow.

may fail to execute correctly for various reasons, such as the unavailability of an atomic web service involved in a plan, or simply because the output of their successful execution is not the expected one, an AI planning algorithm used should take into consideration the non-determinism of the domain.

For this purpose, we are currently considering the incorporation of a contingent planner (Hoffmann and Brafman 2005; Weld, Anderson and Smith 1998), so as to generate plans that can cope with the most influential and likely contingencies. Our goal is not to develop a plan for every possible contingency, as the WSC domain has too many sources of uncertainty for such an approach to succeed.

The approach we plan to follow consists of the use of a complete algorithm, such as A* (Hart, Nilsson and Raphael 1968), to generate all the possible plans in a given period of time, starting from an optimal one, with an increasing cost, until we reach a time limit set by the application's user. A - suboptimal – contingency plan can be constructed by linking these plans by searching for natural join points, i.e., when search nodes share a predecessor through different sets of outcomes, and by removing any plans that contain redundant actions, i.e., repetitive actions or ones that do not produce any useful results. A somewhat similar approach using GraphPlan (Blum and Furst 1997) is used in (Little 2006).

Finally, we will convert the PPDDL plan back to an OWL-S (composite) web service, that is, create an OWL-S profile and its process description, without, however, providing a corresponding WSDL definition, in a fashion analogous to that described in (Kona et al. 2008; Ziaka, Vrakas and Bassiliades 2011).

In short, the profile description of the new composite web service will treat it as an atomic service with IOPEs, while the process model will be based on OWL-S control constructs that describe the way the web services that compose the composite one interact with each other. The OWL-S API (University of Basel - Database and Information Systems Group 2012) that will be used to implement the conversion supports composite processes that use OWL-S control constructs, such as ⟨*Split+Join*⟩, and conditional constructs like ⟨*If-Then-Else*⟩, which will be necessary to produce correct solutions to the use cases already presented.

Future Research Ideas

An example of a practical use of web services in real life is the case of the well known "virtual travel agency"; in such a scenario, a user would require to book his tickets to travel to and from a specific destination, as well as his hotel and means of transportation there. Since there is an extensive collection of alternative options for each of these aspects, e.g., airplane or train tickets, different hotels at each destination or various car leasing companies, and a correct combination of them in a manner that results in a minimum cost is difficult, an automated solution would be beneficent.

This scenario, as well as other everyday use cases, such as the booking of concert tickets, the purchase of different parts of a PC, or the comparison of prices of a specific product in several websites, could be efficiently added in a personal time management system by its users as tasks that can be achieved automatically through composite web services. Our aim is to incorporate the system described in the previous section in such a platform, namely SELFPLANNER (Refanidis and Alexiadis 2011). The fact that the final composite web services will take into account the non-determinism in the domain makes our approach even more suitable for such a purpose, as it will be fault-resilient, requiring less or none intervention from the end user.

Conclusion

The work described here stems from the fact that the recent bibliography suggests a lack of real world implemented web service composition systems, as well as a gap in the evaluation of existing ones. To the best of our knowledge, the final system will be the first online application of its kind able to support various stages of Web Service Composition.

Moreover, we have already presented an evaluation framework based on pre-defined scenarios, quantitative criteria, and a comparison between manual and automatic WSC, as well as a standard test set. It is our hope that it can be used by other WSC works as a common test bed, as they provide detailed descriptions of the web services used and their intended goals, and can be used by systems supporting either deterministic or non-deterministic planning.

Our current and future work that treats web service composition as an inherently non-deterministic process and aiming to apply such an approach in a practical real world application is unique. However, several problems can arise in such an effort, the most important being the lack of real world freely available semantic web services that can be used efficiently in a personal time management system. Such problems can be tackled through the use of dummy tailored made WSs or simple use case scenarios, specifically designed to conform to the actual availability of real world semantic web services.

We expect that in the near future we will be able to demonstrate the first results of this effort through a publicly available online prototype.

Acknowledgements

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: Heracleitus II. Investing in knowledge society through the European Social Fund.

References

Blum, A.; and Furst, M. 1997. Fast planning through planning graph analysis. *Artif. Intell.* 90:281–300.

Chan, M.; Bishop, J.; and Baresi, L. 2007: Survey and comparison of planning techniques for web services composition, Technical Report, University of Pretoria.

Clement, L.; Hately, A.;von Riegen, C.; and Rogers, T. 2004. UDDI version 3.0.2, http://uddi.org/pubs/uddi_v3.htm

EBM Websourcing 2012. Petals BPM architecture overview, http://research.petalslink.org/display/petalsbpm/Architecture+overview

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci., Cybern.* 4 (2): 100–107.

Hatzi, O.; Vrakas, D.; Nikolaidou, M. et al., 2011. An integrated approach to automated semantic web service composition through planning, *IEEE Trans. Service Computing* 301-308.

Hoffmann J.; and Brafman, R. 2005. Contingent planning via heuristic forward search with implicit belief states. In *Proceedings of the 15th International Conference on Automated Planning and Scheduling* (ICAPS '05), 71-80.

Hoffmann, J.; Bertoli, P.; M. Helmert; and M. Pistore 2009. Message-based web service composition, integrity constraints, and planning under uncertainty: a new connection. *J. Artif. Intell. Res.* 35:49-117.

Klusch, M.; Gerber, A.; and Schmidt, M. 2005. Semantic web service composition planning with OWLS-Xplan. In *Proceedings of the First International AAAI Fall Symposium on Agents and the Semantic Web*.

Klusch, M.; Gerber, A.; and Schmidt, M. 2005. Semantic web service composition planning with OWLS-Xplan. In *Proceedings of the First International AAAI Fall Symposium on Agents and the Semantic Web*.

Kona, S.; Bansal, A.; Blake, M.B.; and Gupta, G. 2008. Generalized Semantics-Based Service Composition. In *Proceedings of IEEE International Conference on Web Services* (ICWS’08).

Kuzu M.; and Cicekli, N. 2012. Dynamic planning approach to automated web service composition. *J. Appl. Intell.* 36:1-28.

Little, I. 2006. Paragraph: A graphplan based probabilistic planner. In *Proceedings of the Fifth International Planning Competition* (IPC-5).

Markou, G. 2012. Heracleitus II - WSC Use Case Scenarios, Technical Report. Department of Applied Informatics, University of Macedonia. http://ai.uom.gr/gmarkou/Files/Mad_Swan_Use_Case_Scenarios.pdf

Markou, G. 2013. Heracleitus II – (P)PDDL Use Case Scenarios, http://ai.uom.gr/gmarkou/Files/(P)PDDL%20domains/

Martin, D.; Burstein, M.; Hobbs, J. et al. 2004. OWL-S: Semantic Markup for Web Services, http://www.w3.org/Submission/OWL-S/

McDermott, D.V. 2002. Estimated-regression planning for interactions with web services. In *Proceedings of the Sixth International Conference on Artificial Intelligence Planning Systems* (AIPS ’02), 204-211.

McIlraith S.; and Son, T. 2002. Adapting Golog for composition of semantic web services. In *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning* (KR2002), 482-496.

Mesmoudi, A.; Mrissa, M.; and Hacid, M. 2011. Combining configuration and query rewriting for web service composition. In *Proceedings of IEEE International Conference on Web Services* (ICWS '11), 113-120.

Object Management Group 2011. Business Process Model And Notation (BPMN) Version 2.0, http://www.omg.org/spec/BPMN/2.0/

Pedrinaci, C.; Liu, D.; Maleshkova, M. et al. 2010. iServe: a linked services publishing platform. In *Proceedings of the Ontology Repositories and Editors for the Semantic Web Workshop at the 7th Extended Semantic Web Conference* (ORES '10).

Refanidis, I.; and Alexiadis, A. 2011. Deployment and Evaluation of SelfPlanner, an Automated Individual Task Management System. *Comput. Intell.* 27(1): 41-59.

S3 Contest 2011. Retrieval performance evaluation of matchmakers for semantic web services, http://www-ags.dfki.uni-sb.de/~klusch/s3/html/2011.html

SemWebCentral 2010. OWL-S Service Retrieval Test Collection, http://semwebcentral.org/frs/?group_id=89

Sirbu, A.; and Hoffmann, J. 2008: Towards Scalable Web Service Composition with Partial Matches. In *Proceedings of the IEEE International Conference on Web Services* (ICWS), 29-36.

University of Basel - Database and Information Systems Group 2012. OWL-S API introduction, http://on.cs.unibas.ch/owls-api/index.html

Weld, D.S.; Anderson, C.R.; and Smith, D.E. 1998. Extending Graphplan to handle uncertainty and sensing actions. In *Proceedings of the 15th National/10th Conference on Artificial Intelligence/ Innovative Applications of Artificial Intelligence* (AAAI/ IAAI '98), 897-904.

Younes, H. L. S.; and Littman, M. L. 2004: PPDDL1.0: An extension to PDDL for expressing planning domains with probabilistic effects, Technical Report CMU-CS-04-167, School Of Computer Science, Carnegie Mellon University.

Ziaka, E.; Vrakas, D.; and Bassiliades, N. 2011. Translating web services composition plans to OWL-S descriptions. In *Proceedings of the Third International Conference on Agents and Artificial Intelligence* (ICAART ’11), 167-176.