Towards Automatic Web Service Discovery and Composition in a Context with Semantics, Messages, and Internal Process Flow (A Position Paper)

Daniela Berardi[1], Diego Calvanese[2], Giuseppe De Giacomo[1], Richard Hull[3], Massimo Mecella[1]

[1] *Università di Roma "La Sapienza"*
berardi@dis.uniroma1.it
degiacomo@dis.uniroma1.it
mecella@dis.uniroma1.it

[2] *Libera Università di Bolzano/Bozen*
calvanese@inf.unibz.it

[3] *Bell Labs Research*
Lucent Technologies
hull@lucent.com

April 29, 2005

Abstract

In this short position paper we briefly describe our recent and on-going work on `Colombo`, a framework in which web services are characterized in terms of *(i)* the atomic processes (i.e., operations) they can perform; *(ii)* their impact on the "real world" (modeled as a relational database); *(iii)* their transition-based behavior; and *(iv)* the messages they can send and receive (from/to other web services and "human" clients). As such, `Colombo` combines key elements from the standards and research literature on (semantic) web services. In particular, `Colombo` complies with key aspects of the emerging Semantic Web Services Initiative (SWSI) Ontology, including representation of atomic processes (that impact an abstraction of the "real world") and messages (for data flow between web services), along with a concrete model of the process and data flow within web services.

Using `Colombo`, we are studying the problem of automatic service discovery and composition (synthesis). In [5, 4], we devise a sound, complete and terminating algorithm for building a composite service (or determining that none exists) under various restrictions. Specifically, this work develops *(i)* a technique for handling the data, which ranges over an infinite domain, in a finite, symbolic way, and *(ii)* a technique to automatically synthesize composite web services, based on Propositional Dynamic Logic. We view this as an important first step in the eventual development of practical techniques for automatic discovery and composition of web services that are described using the emerging SWSI ontology.

Service Oriented Computing (SOC [1]) is the computing paradigm that utilizes web services (also called *e*-Services or, simply, services) as fundamental elements for realizing distributed applications/solutions. Web services are self-describing, platform-agnostic computational elements that support rapid, low-cost and easy composition of loosely coupled distributed applications.

SOC poses many challenging research issues, the most prominent being *web service composition*. Web service *composition* addresses the situation when a client request cannot be satisfied by any available service, but by suitably combining "parts of" available services. Composition involves two different issues [1]. The first, typically called *composition synthesis*, is concerned with synthesizing a specification of how to coordinate the component services to fulfill the client request. Such a specification can be produced either *automatically*, i.e., using a tool that implements a composition algorithm, or *manually* by a human. The second issue, often referred to as *orchestration*, is concerned with how to actually achieve the coordination among services, by executing the specification produced by the composition synthesis and by suitably supervising and monitoring both the control flow and the data flow among the involved services. Research on orchestration draws from several other research areas, including most notably workflow, and has already produced the BPEL standard.

In the research discussed here, we have been studying the problem of automatic composition synthesis of web services. Within this area there are two basic approaches to synthesis: for *single-use* and for *multiple use*. In the single-use case (e.g., see [15]), a client specifies a desired service behavior. The composition synthesis algorithm creates a composite service from existing services, and that service is executed exactly once to accomplish the desired service behavior. The next time that this client or some other wants a (perhaps similar) behavior, the composition synthesis algorithm is used again to create a composite service. In the multiple use case (e.g., see [6]), the desired service behavior is specified as a (re-usable) transition system (e.g., as a generalized automaton of some kind). The composition synthesis algorithm is run once to create a composite service, and the resulting service can be executed many times, for many clients. In the research described here, we focus on composition synthesis of services for multiple use.

In our research [5, 4] we have introduced an abstract model, called `Colombo`, that combines four fundamental aspects of web services, namely:

(a) A world state, representing the "real world", viewed as a database instance over a relational database schema, referred to as world schema. It represents the "real world" This is similar to the family of "fluents" found in semantic web services models such as OWL-S [13], and more generally, found in situation calculii [16].

(b) Atomic processes (i.e., operations), which can access and modify the world state, and may include conditional effects and non-determinism. These are inspired by the atomic processes of OWL-S.

(c) Message passing, including a simple notion of ports and links, as found in web services standards (e.g., WSDL [3], BPEL4WS [2]) and some formal investigations (e.g., [7, 10]).

(d) The behavior of web services (which may involve multiple atomic processes and message-passing activities) is specified using finite state transition system, in the spirit of [6, 7, 10].

The first three elements parallel in several aspects the core elements of the emerging SWSL (Semantic Web Service Language) ontology for semantic web services (e.g., see [19] for the latest output from this effort). The fourth element provides an abstract approach to formally model the internal process model of a web service. (The SWSL ontology can support many forms of internal process model for services; an automata-based approach is one of these.) With regards to the relational database, it is typical with `Colombo` that some individual relations will be accessible only by one service, and others might be accessible only by a subset of services. In this way we can represent the situation where different services have sole or overlapping access to different databases. The third and fourth elements above provide one approach to specifying the sequencing or behavioral patterns of a service involved with message passing. As such, these have some relationship to the concepts "orchestration" and "choreography" as used in the Web Service Modeling Ontology (WSMO) [8], but that relationship has not yet been explored in the context of `Colombo`. We also recall from [11] that Service Oriented Computing can play a major role in transaction-based data management systems, since web services can be exploited to access and filter data. Through its use of a relational database the `Colombo` framework provides one approach for exploring this in a formal setting.

We also assume that:

(e) Each web service instance has a "local store", used to capture parameter values of incoming messages and the output values of atomic processes, and used to populate the parameters of outgoing messages and the input parameters of atomic processes. Conditional branching in a web service will be based on the values of the local store variables at a given time. (The conditions in atomic process conditional effects are based on both the world state and the parameter values used to invoke the process.)

(f) Finally, we introduce a simple form of integrity constraints on the world state.

A client of a web service interacts with it by repeatedly sending and receiving messages, until a certain situation is reached. In other words, also the client behavior can be abstractly represented as a transition system.

In order to address the problem of automatic web service composition, we introduce the notion of "goal service", denoting the behavior of a desired (re-usable) composite service: it is specified as a finite state transition-based web service, that interacts with a client and invokes atomic processes. Our challenge is to build a mediator, which uses messages to interact with pre-existing web services (e.g., in an extended UDDI directory), such that the overall behavior of the mediated system faithfully simulates the behavior of the goal service. (In the terminology of [1] the mediator performs orchestration of the pre-existing services.)

The contribution of this work to date is multifold: *(i)* `Colombo` unifies and extends several of the most important frameworks for services and service composition; *(ii)* it presents a technique to reduce infinite data value to finite symbolic data (generalizing a technique of [12]); *(iii)* it exploits and extends techniques based on Propositional Dynamic Logic to automatically synthesize a composite service (generalizing the approach developed in [6]), under certain assumptions; *(iv)* it provides an upper bound on the complexity of this problem. To the best of our knowledge, the work reported in [5, 4] is the first one proposing an algorithm for web service composition where web services are described in terms of *(i)* atomic processes, *(ii)* transition-based process models, *(iii)* their impact on a database representing the "real world", and *(iv)* message-based communication.

Selected Related Work

BPEL4WS [2] allows for (manually) specifying the coordination among multiple web services, expressed in WSDL. The data manipulation internal to web services is based on a "blackboard approach", i.e., a set of variables that are shared within each orchestration instance. Thus, on the one hand BPEL4WS provides constructs for dealing with data flow, but on the other hand, it has no notion of world state.

OWL-S [13] is an ontology language for describing semantic web services, in terms of their inputs, outputs, preconditions and (possibly conditional) effects, and of their process model. On the one hand OWL-S allows for capturing the notion of world state as a set of fluents, but on the other hand it is not clear how to deal with data flow (within the process model).

Several works on automatic composition of OWL-S services exists, e.g., [14, 17]. Most results are based on the idea of *sequentially* composing the available web services, which are considered as black boxes, and hence atomically executed. Such an approach to composition is tightly related to Classical Planning in AI. Consequently, most goals express conditions on the real world, that characterize the situation to be reached: therefore, the automatically devised composition can be exploited only *once*, by the client that has specified the goal. Conversely, in `Colombo` the goal is a specification of the *transition system* characterizing the process of a desired composite web service. Thus, it can be *re-used* by several clients that wants to execute that web service.

`Colombo` extends the Roman model, presented in [6], mainly by introducing data and communication capabilities based on messages. The level of abstraction taken in [6] focuses on (deterministic, atomic) actions, therefore, the transition system representing web service behavior is deterministic. Also, all the interactions are carried out through action invocation, instead of message passing. Finally, in [6] there is no difference between the transition system representing the client behavior and the one specifying the goal, as it is in `Colombo`.

`Colombo` has its root also in the Conversation model, presented in [7, 10], extending it to deal with data and atomic processes. Web services are modeled as Mealy machines (equipped with a queue) and exchange sequence of messages of given types (called conversations) according to a predefined set of channels. It is shown how to synthesize web services as Mealy machines whose conversations (across a given set of channels) are compliant with a given specification. In [10] an extension of the framework is proposed where services are specified as guarded automata, having local XML variables in order to deal with data semantics.

In [18] web services (and the client) are represented as possibly non-deterministic transition systems, communicating through messaging, and composition is achieved exploiting advanced model cheking techniques. However, a limited support for data is present and there is no notion of local store. It would be interesting to attempt to apply in their framework our techiques for finitely handling data ranging over an infinite domain, in order to provide an extension to it.

Finally, it is interesting to mention the work in [9], where the authors focus on data-driven services, characterized by a relational database and a tree of web pages. In such a framework, the authors study the automatic verification of properties of a single service, which are defined both in a linear and in a branching time setting.

Acknowledgement

The authors would like to thank Maurizio Lenzerini, Jianwen Su and the members of the SWSL working group for valuable discussions concerning their work on `Colombo` and its applications.

References

[1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services. Concepts, Architectures and Applications*. Springer, 2004.

[2] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana. Business Process Execution Language for Web Services (BPEL4WS) -Version 1.1. `http://www-106.ibm.com/developerworks/library/ws-bpel/`, 2004.

[3] Ariba, Microsoft, and IBM. Web Services Description Language (WSDL) 1.1. Available on line: `http://www.w3.org/TR/2001/NOTE-wsdl-20010315`, 2001.

[4] D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, and M. Mecella. Automatic composition of web services in Colombo. In *Proc. of 13th Itallian Symp. on Advanced Database Systems*, June 2005.

[5] D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, and M. Mecella. Automatic Composition of Transition-based Semantic Web Services with Messaging. Technical report, Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Roma, Italy, April, 2005.

[6] D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Mecella. Automatic Composition of *e*-Services that Export their Behavior. In *Proceedings of the 1st International Conference on Service Oriented Computing (ICSOC 2003)*, volume 2910 of *LNCS*, pages 43–58. Springer, 2003.

[7] T. Bultan, X. Fu, R. Hull, and J. Su. Conversation Specification: A New Approach to Design and Analysis of E-Service Composition. In *Proceedings of the 12th International World Wide Web Conference (WWW 2003)*, pages 403–410. ACM, 2003.

[8] J. de Bruijn, C. Bussler, J. Domingue, D. Fensel, M. Hepp, M. Kifer, B. König-Ries, J. Kopecky, R. Lara, E. Oren, A. Polleres, J. Scicluna, and M. Stollberg. Web Service Modeling Ontology (WSMO). Technical report, DERI, 2005.

[9] A. Deutsch, L. Sui, and V. Vianu. Specifi cation and Verifi cation of Data-driven Web Services. In *Proceedings of the 23nd ACM SIGACT SIGMOD SIGART Symposium on Principles of Database Systems (PODS 2004)*, pages 71–82. ACM, 2004.

[10] X. Fu, T. Bultan, and J. Su. Analysis of interacting BPEL web services. In *Proceedings of the 13th International World Wide Web Conference (WWW 2004)*, pages 621–630. ACM, 2004.

[11] P. Helland. Data on the outside versus data on the inside. In *CIDR*, pages 144–153, 2005.

[12] R. Hull and J. Su. Domain independence and the relational calculus. *Acta Informatica*, 31(6):513–524, 1994.

[13] D. Martin, M. Paolucci, S. McIlraith, M. Burstein, D. McDermott, D. McGuinness, B. Parsia, T. Payne, M. Sabou, M. Solanki, N. Srinivasan, and K. Sycara. Bringing Semantics to Web Services: The OWL-S Approach. In *1st International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004)*, 2004.

[14] S. McIlraith, T. Son, and H. Zeng. Semantic Web Services. *IEEE Intelligent Systems*, 16(2):46 – 53, 2001.

[15] S. Narayanan and S. McIlraith. Simulation, Verifi cation and Automated Composition of Web Services. In *Proceedings of the 11th International World Wide Web Conference (WWW 2002)*, pages 77 – 88. ACM Press, 2002.

[16] R. Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, 2001.

[17] E. Sirin, B. Parsia, D. Wu, J. Hendler, and D. Nau. HTN planning for Web Service composition using SHOP2. *J. Web Sem.*, 1(4):377–396, 2004.

[18] P. Traverso and M. Pistore. Automated Composition of Semantic Web Services into Executable Processes. In *Proceedings of the Third International Semantic Web Conference*, pages 380–394, 2004.

[19] S. W. S. L. working group. Swsl home page. `http://www.daml.org/services/swsl/`, 2005.