Experiences with OWL-S, Directions for Service Composition: The Cashew Position

Barry Norton[1] ⋆

Knowledge Media Institute, Open University, Milton Keynes, UK
b.j.norton@open.ac.uk

Abstract. Having used OWL-S for some time, both in the form of the 'Virtual Machine' [12] and our own experimental implementation [7], the Cashew project has drawn a number of conclusions about its existing process model which we assert as follow:
1. OWL-S is not service composition;
2. OWL-S defines orchestration but not choreography;
3. OWL-S is incompatible with choreography;
4. OWL-S is insufficiently composable;
5. OWL-S needs compositional semantics.

This position paper will details, and attempt to justify, these assertions and sketch the Cashew approach to their solution.

OWL-S may be said to concern the use of 'semantics', in the sense of the *Semantic* Web [2], to describe services, in the sense of Web *Services* [16]. This involves the use of ontologies to describe the concepts dealt with by services, but also extends to the *static* (*i.e.* structural) description of service composition.

On the other hand, 'semantics', as usually understood by computer scientists - where formal semantics are given to computation, a dynamic, rather than a static, paradigm - has only begun to be applied[1]. We shall call this latter form of semantics *'behavioural semantics'*. During its early incarnation in DAML-S, the basis of the OWL-S process model received two forms of behavioural semantics: one oriented towards process calculus style [1], and one in Petri net style [10]. Missing, however, was a concern for the principal of *compositionality* - that the *semantics for a composition should derive directly from the individual semantics of the component parts* (and not a recomputation of these).

The need for compositionality in semantics for the Semantic Web in general is already acknowledged [13] [14]. The reason compositionality is such an important property of semantics is that this allows a semantic model to be built by composition alongside the definition of syntax, for instance in an interactive editor, as well as being the enabling property of semantics for modular analysis.

⋆ This work is supported by the DIP project, an Integrated Project (no. FP6 - 507483) supported by the European Union's IST programme.

[1] While reasoning might involve dynamics, for instance in logic programming or in theorem proving, formal semantics are only given to the logic, not the computation.

The first major result of the Cashew project has been to define a compositional semantics for OWL-S[2] [11]. Rather than define a formal behavioural semantics *directly* for OWL-S process models, we chose to go through two intermediate levels. The first is an equivalent language to the process model ontology, but with more scope for composition (hence, in fact, making compositionality a more difficult principal to provide for). In particular, it is necessary in OWL-S to define all of the incoming dataflow for the performance of a 'process' at the point the performance is declared (as part of a composite process). In our intermediate *workflow* language, connections in the dataflow become a first-class constructs so that these can be introduced (composed) separately. This is a more natural model for interactive editing. We shall see that this workflow language becomes one of the bases for our formal language *Cashew-S* for service composition.

The second intermediate in our semantic translation is *Cashew-Nuts*, a true *process* language. This is a process calculus defined by extension of CCS [9] with multi-party synchronisations with an implicit notion of priority. As in CCS its primary behavioural semantics are given in the form of labelled transition systems via structured operational semantics. Compositionality through strong bisimulation — but also a 'temporal observation congruence', which allows abstraction from internal states and communications — has been established.

Having discussed the last two assertions, and our existing work, we now move on to make our position statement as embodied in the first three assertions. The WSMO group have proposed that the behavioural models for each semantic web service should consist of both an 'orchestration' and a 'choreography', that is to say an internal and external view of its composition [15][3]. Assertion 2 is by now well accepted and has been discussed by other authors.

What seems not to have been documented is how a lack of fit with the general view of services, asserted as 1, affects the ability to adopt choreography into the OWL-S world. In the 'Web Services Architecture', services in general offer multiple operations [16]. The interpretation given to the processes defined in OWL-S is that each atomic process is an operation. Similarly, the orchestration forms only one operation. Consequently we could say that OWL-S process models really define 'operation composition' and do not operate at the service level.

The problem here is that (one of) the aim(s) of (WSMO's) choreography is to establish a 'protocol' by which a client's session with a service can invoke its operations. In the commonly used bookshop example, it might be necessary to log in before a purchase can be made. Even though an OWL-S orchestration could involve both the 'log in' and 'purchase' operation, it would not be made clear that these belonged to the same session (one orchestration could operate across more than one session, for instance in mediating between two accounts).

[2] Note that it makes no sense to ask whether OWL-S is compositional or not — this is a semantic property that applies to a given *semantics* for OWL-S.

[3] In this, the use of the term choreography, which is not novel or original to WSMO, is slightly different to other definitions, such as W3C's, where a *multi-party* conversation, to achieve some particular task, is implied.

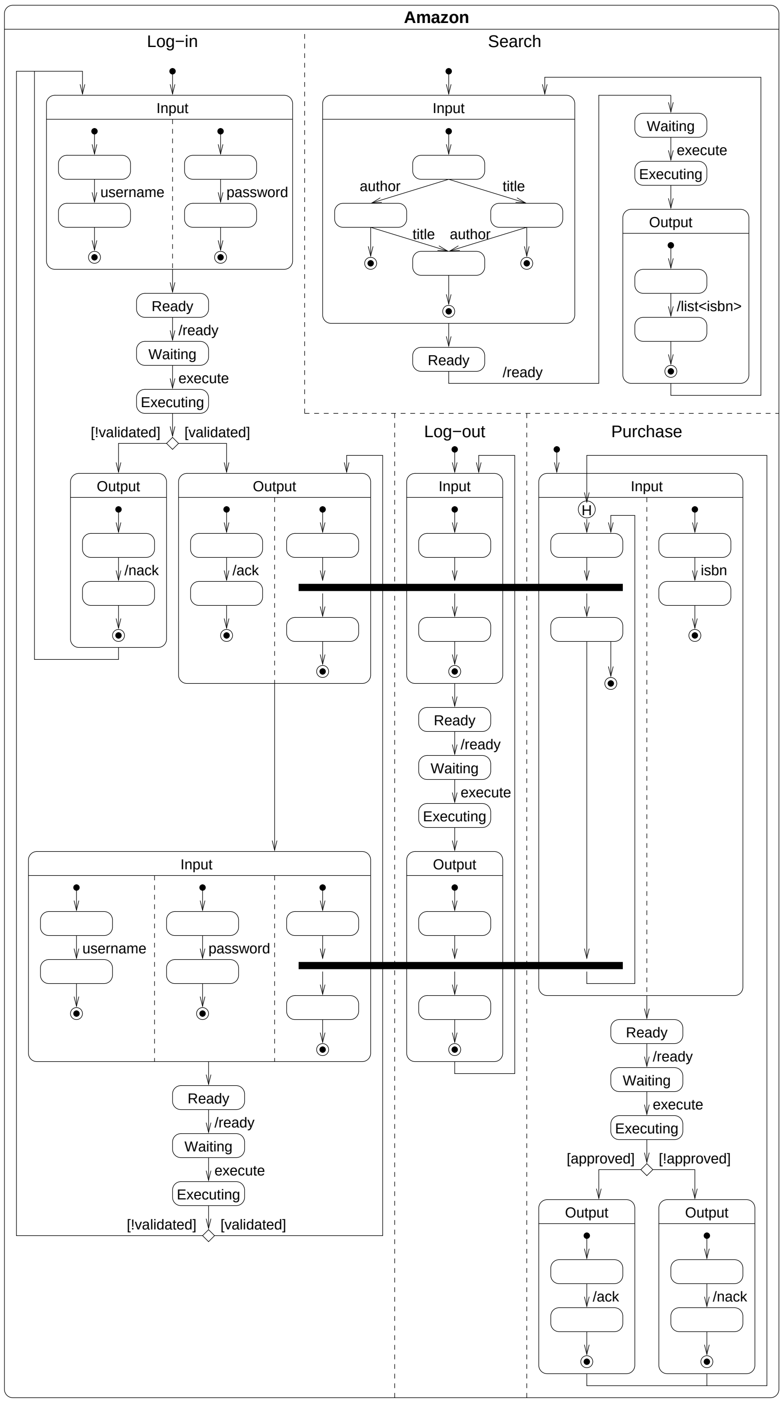
IRS-III [3], being one of the two reference implementations for WSMO, tackles this aim by splitting two separate notions of choreography. A deployed service choreography describes interactions with all permitted clients from the service's point of view. The diagram on the following page illustrates many of the features we should like to express in these choreographies. It shows not only that purchase is only possible in a 'logged-in' state, but that logging out is also only possible in this state and rules out purchase until another log-in. Meanwhile searches are possible regardless of log-in, and also demonstrate optional inputs — a client can search on a title, an author or (but not necessarily) both.

On the other hand, client choreographies, formalised in [4], are like W3C choreographies in expressing a conversation that achieves a particular task. The difference is that only a single client is involved, and the interaction is expressed from that client's point of view. That client may ultimately be an orchestration, so that the messages are produced and consumed are routed to and from multiple other parties, but this is arranged separately from the client choreography in order that the choreography can be reused. The means for this reuse is the creation of abstract goals, expressed in an ontology, which are mapped as being realised by client choreographies.

In this way the coordination between low-level operations, and the dependencies between them (as expressed in the deployed service's choreography), is isolated from the workflow. IRS-III can be viewed as a broker allowing an abstract orchestration of goals to be built, to achieve some high-level task. At run-time IRS locates client choreographies by which the goals can be realised.

In the current implementation, client choreographies are directly expressed in abstract state machines [5], and a fragment of OWL-S has been used as the basis of orchestration of goals [8]. The intention of the Cashew project is to show how a high-level ontology can be created for orchestrations, generalising on OWL-S, as well as the choreographies of deployed services and their clients. Furthermore, we shall show how the basic ideas of using UML Activity Diagrams to illustrate workflows, for instance implemented in [6], can be extended, how client choreographies can also be illustrated in Activity Diagrams, and service choreographies can be illustrated in State Machine Diagrams, as shown on the following page.

Most importantly, we shall show how our compositional OWL-S semantics can be built upon to give an operational semantics to all these models, and how these semantics can be grounded in abstract state machines for compatibility with WSMO. In this way we shall create a full implementation for orchestration within the IRS and, furthermore, have a formal model on which conformance properties can be checked.


Amazon
Log−in
Search
Input
username
password
Ready
/ready
Waiting
execute
Executing
[!validated]
[validated]
Output
/nack
/ack
author
title
/list<isbn>
Log−out
Purchase
H
isbn
[approved]
[!approved]

References

1. A. Ankolekar, F. Huch, and K. Sycara. Concurrent execution semantics for DAML-S with subtypes. In *Proc. 1st International Semantic Web Conference (ISWC)*, 2002.
2. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):35–43, 2001.
3. J. Domingue, L. Cabral, F. Hakimpour, D. Sell, and E. Motta. IRS-III: A platform and infrastructure for creating WSMO-based semantic web services. In *Proc. of the Workshop on WSMO Implementations (WIW 2004)*, volume ISSN 1613-0073. CEUR Workshop Proceedings, 2004.
4. J. Domingue, S. Galizia, and L. Cabral. Choreography in irs-iii: Coping with heterogeneous interaction patterns in web service. In *Proc. 4th Intl. Semantic Web Conference*, 2005. to appear.
5. E. Börger. High level system design and analysis using abstract state machines. In *Current Trends in Applied Formal Methods (FM-Trends 98)*, number 1641 in LNCS, pages 1–43. 1999.
6. D. Elenius, G. Denker, D. Martin, F. Gilham, J. Khouri, S. Sadaati, and R. Senanayake. The OWL-S editor - a development tool for semantic web services. In *Proc. 2nd European Semantic Web Conf.*, number 3532 in LNCS, 2005.
7. S. Foster, A. Hughes, and B. Norton. Composition and semantic enhancement of web services: The CASheW-S project. In *Proc. 1st Young Researchers' Workshop on Service-Oriented Computing (YR-SOC'05)*, 2005.
8. F. Hakimpour, J. Domingue, E. Motta, L. Cabral, and Y. Lei. Integration of OWL-S into IRS-III. In *Proc. 1st AKT Workshop on Semantic Web Services*, volume 122. CEUR Workshop Proceedings, 2004.
9. A. J. R. G. Milner. *Communication and Concurrency*. Prentice Hall, 1989.
10. S. Narayanan and S. McIlraith. Simulation, verification and automated composition of web services. In *Proc. 11th Intl. Conf. on World Wide Web (WWW 2002)*, 2002.
11. B. Norton, S. Foster, and A. Hughes. A compositional semantics for OWL-S. In *Proc. 2nd Intl. Workshop on Web Services and Formal Methods (WS-FM 05)*, number 3670 in LNCS, Sept 2005.
12. Massimo Paolucci, Anupriya Ankolekar, Naveen Srinivasan, and Katia Sycara. The DAML-S virtual machine. In *Proc. 2nd Intl. Semantic Web Conference (ISWC2002)*, volume 2870 of *LNCS*, pages 290–305. Springer Verlag, 2003.
13. A. Sheth, C. Ramakrishnan, and C. Thomas. Semantics for the Semantic Web: The implicit, the formal and the powerful. *Intl. Journal on Semantic Web and Information Systems*, 1(1):1–18, 2005.
14. M. Solanki, A. Cau, and H. Zedan. Augmenting semantic web service description with compositional specification. In *Proc. 13th Intl. World Wide Web Conf. (WWW 2004)*, 2004.
15. M. Stollberg and D. Fensel. Ontology-based choreography and orchestration of WSMO services. http://www.wsmo.org/TR/d14/, July 2005.
16. W3C. Web services architecture. http://www.w3.org/TR/ws-arch/, Feb 2004.