An Agent-Based Compositional Framework

R. Anane[1], Y. Li[2,*], C-F Tsai[3], K-M Chao[1], and M. Younas[1]

[1] DSM Research Group, School of MIS, Coventry University, UK
{r.anane, k.chao, m.younas}@coventry.ac.uk
[2] Software School, Fudan University, Shanghai, P.R. China
liys@fudan.edu.cn
[3] Department of IM, Aletheia University, Taiwan
tsai@email.au.edu.tw

Abstract. Web services requirements for means of automating the coordination of distributed heterogeneous applications have been addressed by a two-pronged initiative. One has been driven by process management technologies, represented by the compositional language BPEL4WS, and the other by the Semantic Web technologies, represented by ontologies and agents. The framework presented in this paper integrates agent technology with BPEL4WS, to enable two levels of coordination, namely local and centralised coordination through composition, and wider, distributed coordination by means of negotiation between agents. This framework takes advantage of the efficient management of processes in BPEL4WS and of the flexibility and versatility of agents. A design and run-time environment, called SOA, has been developed to evaluate the feasibility of the framework.

1 Introduction

The execution of distributed applications, in general, and Web services, in particular, requires coordination. In the process management approaches, this has seen the development of compositional languages for workflow management, such as BPEL4WS. The underlying models are relatively efficient and their behaviour is predictable. Interactions take place in a well-defined and stable environment, with clearly specified components; coordination is by workflow management. They implement a static binding policy, and operate at the syntactic level. In Semantic Web technologies, on the other hand, the aim is to go beyond mere interoperation towards increased automation. Of importance in this context is the creation of semantically rich entities that can display intelligent behaviour and engage in meaningful negotiation [1].

These two forms of coordination, namely workflow management and negotiation, are complementary and can be successfully integrated by adding semantics to the composition process of Web services [2]. One key feature of this work of integration is the dynamic discovery and utilisation of Web services. As an example of integration, wrappers were used to make Web services behave like agents [3]. The

[*] Corresponding author.

Y. Zhang et al. (Eds.): APWeb 2005, LNCS 3399, pp. 579–584, 2005.

work presented in this paper is concerned with the development of a framework based on a symbiotic relationship between agents and Web services, within a BPEL4WS platform. The aim is to reconcile the two forms of coordination, so that Web services can be composed, discovered dynamically, reasoned upon and executed efficiently by workflow management.

The rest of the paper is organised as follows. Section 2 defines the main issues in Web service composition. Section 3 gives an introduction to Semantic Web technologies and agents. Section 4 presents a conceptual view of the proposed framework. Section 5 deals with the implementation of the system. Section 6 puts the framework in context, and Section 7 concludes the paper.

2 Web Services and BPEL4WS Composition

One consequence of the decoupling of ownership of software from its use in Web services is that the binding of services can be performed dynamically. Web services are further enhanced by the introduction of formalisms for composing them into new Web services, which can be discovered and invoked in response to user requirements [4]. BPEL4WS is a workflow-based composition language for describing interactions of Web services as business processes [5]. A new composite Web service can be generated from the aggregation of other Web services. Its interface can be described as a set of WSDL *PortTypes*, in the same manner as for atomic Web services.

BPEL4WS and its engine, BPWS4J, offer predicable behaviour and performance. BPEL4WS has, however, some limitations, in particular its centralised workflow enactment and the fact that Web services must be known and defined *a priori* [6]. The main criticism levelled at these technologies is that they operate at the syntactic level, are implementation focused and require human intervention at various stages [7]. In contrast, the Semantic Web technologies underline the fact that the drive towards more automation and meaningful interaction between Web services requires greater semantic content in the descriptions.

3 Semantic Web Technologies and Agents

The semantic gap between XML-based constructs and agents can be bridged by use of Semantic Web technologies, such as OWL-S [8]. The aim of the Semantic Web initiative is to provide technologies that will enable heterogeneous systems to collaborate in the execution of an activity. For Web services description, the introduction of OWL-S is a significant factor in matching service providers and service requestors [9]. OWL-S is an ontology for providing richer Web service description, and has three components:

1. ServiceProfile: describes what the service does, its inputs and outputs and its preconditions and effects (IOPE); this is equivalent to UDDI content.
2. ServiceModel: describes how the service works (control and dataflow in its use). This is similar to BPEL4WS.
3. ServiceGrounding: describes how the service is implemented and provides a mapping from OWL-S to WSDL.

Agents are suitable for highly dynamic environments and operate at a conceptual level. BDI agents [10] are particularly apt at exploiting the semantically rich environment defined by OWL-S ontologies [11]. These agents hold beliefs (B), have goals (D) and use Intentions/plans (I) to achieve their goals. An agent can be generated from an OWL-S structures. The ServiceProfile in OWL-S maps to an agent's beliefs (B). The ServiceModel is mapped to a set of intentions associated with plans (I). Preconditions and effects from the ServiceProfile will translate into conditions and effects for the BDI plan. The desire (D) is specified by additional functionality in conjunction with the specification of the ServiceProfile. In essence, the ServiceProfile and the ServiceModel in OWL-S provide the semantics, while the Service Grounding is used to generate the interface signatures.

4 A BPELWS Compositional Framework with Agents

The framework is defined at two levels: the specification of compositions and enactment of the process. The first level concerns the composition process and introduces virtual Web services (VWS) as potential partners, alongside ordinary, concrete Web services. Virtual Web services decouple the composition process from the binding of Web services. A virtual Web service specifies its input and output requirements and associates itself with a nominal *PortType*; its is used in the composition in the same manner as an ordinary Web service. Composition within this framework involves incorporating concrete Web services when known and statically bound, and virtual Web services when unknown and to be dynamically bound.

The second level is identified with the binding of the VWS, and therefore with the enactment of the BPEL4WS process. This requires a mechanism for implementing late or dynamic binding. This form of binding is performed by an agent, which operates on the high-level semantics provided by OWL-S. At design time, two stages are required in order to generate an agent. A VWS, mainly identified by its input/output (I/O), is first augmented with richer semantics provided by OWL-S; the VWS is now endowed with IOPE properties. A BDI agent is subsequently generated from the OWL-S description. The agent obtains the semantic description and combines it with its reasoning mechanism. Once the agent is created from the OWL-S structures it acts as a proxy for the VWS; the OWL-S description is kept for documentation and reasoning purposes. The role of this agent is akin to that of a broker, since it takes requests from a VWS, performs the necessary matchmaking tasks, and then invokes the corresponding Web service on behalf of the VWS.

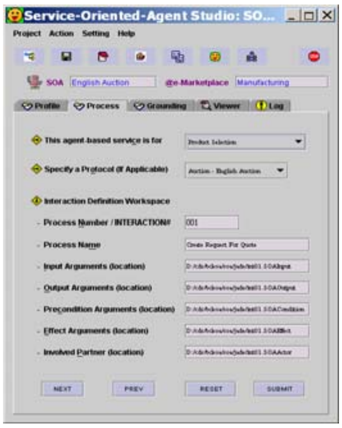
5 Implementation

The implementation involves the provision of an appropriate structure for the VWS, a mechanism for generating an agent from an OWL-S description and a specification of the interaction between the virtual Web service and the agent. A VWS includes operators that trigger events in the agent when it receives the request from the BPWS4J engine, or forwards an input to BPWS4J when the agent relays the required input from other agents or Web services. The VWS contains only interfaces to agents.

In order to facilitate the process of creating service-oriented agents, a template agent was designed, with two elementary functions: one for listening to events triggered by a VWS, and the other one for passing the response from other agents to the VWS. The required input and output must be explicitly described in the agent. The template agent also includes a number of generic coordination protocols such as contract net, English auction, reverse auction etc. A matchmaking mechanism is required for discovering appropriate Web services, according to syntactic signatures and semantic requirements of the composition process. The auction coordination protocol is used to coordinate the selection process.

We have implemented a system with a flexible GUI, in order to support the architectural framework presented above. The system, SOA (Service-Oriented Agent) 1.2 studio [12] includes a template for creating VWS and a template for creating agents (see Figure 1). A GUI was designed so that users can describe semantics of Web services and incorporate OWL-S descriptions. JAXRPC 1.3 and Tomcat are used to support the Web services. For composition the IBM BPWS4J 2.1 engine and BPEL4WS Editor Eclipse plug-in provide an environment for creating and interpreting workflows.

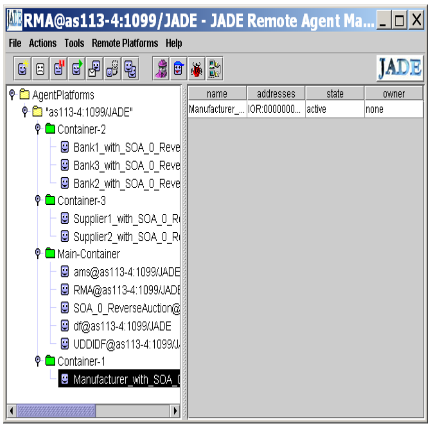
Agent technology was incorporated by means of two components. JADE agent is used for reasoning and communication, while OWL JessKB [13] provides the reasoning capability over OWL-S profiles in the agent. Once the users specify the required information, the studio allows the generation and compilation of code to take place in the same environment. Figure 1 show a GUI provided by SOA for inputting the ServiceProfile information in OWL-S. The user can also specify the ServiceGrounding profile, which can be stored as a project; essential Java code can then be generated, compiled and executed. The SOA creates agents and deploys them, as shown in Figure 2.



SOA: English Auction
e-Marketplace: Manufacturing

This agent-based service is for: Product Selection
Specify a Protocol (If Applicable): Auction - English Auction
Interaction Definition Workspace
- Process Number / INTERACTION#: 001
- Process Name: Create Request For Quote
- Input Arguments (location): [illegible]
- Output Arguments (location): [illegible]
- Precondition Arguments (location): [illegible]
- Effect Arguments (location): [illegible]
- Involved Partner (location): [illegible]

Fig. 1. Model and Profiles in SOA



```
AgentPlatforms
  "as113-4:1099/JADE"
    Container-2
      Bank1_with_SOA_0_Reve
      Bank3_with_SOA_0_Reve
      Bank2_with_SOA_0_Reve
    Container-3
      Supplier1_with_SOA_0_R
      Supplier2_with_SOA_0_R
    Main-Container
      ams@as113-4:1099/JADE
      RMA@as113-4:1099/JADE
      SOA_0_ReverseAuction@
      df@as113-4:1099/JADE
      UDDIDF@as113-4:1099/J
    Container-1
      Manufacturer_with_SOA_(
```

name	addresses	state	owner
Manufacturer_...	IOR:0000000...	active	none

Fig. 2. Agent Platform in SOA

Since the proposed system is to provide an integrated environment for developing agent-based Web services, usability, as a criterion for evaluation, acquires special significance. Users can take advantage of the templates that the system provides for the creation of agents and their coordination. They can easily enhance Web services with semantics through a user friendly GUI. The system combines input from users and templates and generates the necessary code. This can reduce design time and ensure consistency.

6 The Framework in Context

In this section an attempt is made at putting the proposed framework in context, by considering two approaches to composition. In [6], the rationale is to move away from the rigidity of workflow enactment of BPEL4WS/BPWS4J to the decentralised and flexible mode of coordination of multi-agent systems. The work is aimed at producing a multi-agent enactment from BPEL4WS composition. This approach offers flexibility and can lead to the optimised use of resources. Its main drawback, however, is the added complexity entailed by a transition from one domain of execution to another. Closer to the work described in this paper is the model presented in [14] where BPEL4WS is enhanced by Semantic Web technologies as a means of overcoming the limitations of BPWS4J. BPEL4WS is extended with a Semantic Discovery Service (SDS), which acts as a proxy between BPWS4J and the potential partners to be located and selected dynamically. All requests to previously selected partners are directed to the SDS, which implements a late-binding policy. This model preserves the original BPEL4WS structure.

Although our work is less ambitious in scope, the model we propose offers more flexibility and customisation because each virtual Web service is associated with an agent. The decentralisation of the discovery process makes the system more reliable and avoids the single point of failure of the SDS. These features may, however, be costly in computational and in storage terms, not least because of the duplication of resources. The model may require a heavier human intervention than the SDS-based model, because of the semantic enrichment. Both models maintain, however, the original composition structure. The approach promoted by the framework allows for an incremental development of composition and combines the predictability of BPEL4WS enactment with the versatility of Semantic Web technologies.

7 Conclusion

The requirement for the provision of mechanisms to support a two-level mode of coordination has led to the development of a hybrid system that capitalises on the efficiency afforded by compositional BPEL4WS platform and on the versatility of Semantic Web and agent technologies. This was achieved by enhancing Web services through a symbiotic relationship with agent technology. This has allowed, on one hand, for the implementation of dynamic binding of Web services and incorporation into BPWS4J, and for Web services to engage, through its manifestation as an agent, in autonomous behaviour and active negotiation. The system is operational and work is currently carried out on further enhancement and integration of agent technology.

References

1. Chen, J-H, Chao, K-M, Godwin, N., Soo, V-W.: Combining Cooperative and Non-Cooperative Automated Negotiations. Information Systems Frontiers, Kluwer Academic Publisher (2005) (to appear)
2. Sirin, E., Parsia, B., Hendler, J.: Filtering and selecting Semantic Web Services with Interactive Composition Techniques, IEEE Intelligent Systems (July/August 2004) 42-49
3. Knoblock, C., Minton, S., Ambie, J. L., Muslea, M., Oh J., Frank, M: Mixed-initiative, multi-source information assistants, In: Proceedings of the World Wide Web Conference, ACM press, New York, NY (2001) 697-707
4. Limthanmaphon, B., Zhang, Y.: Web Service Composition with Case-Based Reasoning. In: 14th Australasian Database Conference (ADC 2003), Adelaide, South Australia (February 2003) 201-208
5. Khalaf, R., Mukhi, N. , Weerawarana, S.: Service-oriented Composition in BPEL4WS. [Online]. Available: http://www2003.org/cdrom/papers/alternate/P768/choreo_html/ p768-khalaf.htm (2003)
6. Vidal, J. M., Buhler, P., Stahl, C.: Multiagent Systems with Workflows. IEEE Internet Computing (January/February 2004) 76-82
7. Richards, D., van, Spunter S., Brazier, F.M.T., Sabou M.: Composing Web Services using and Agent Factory. In: AAMAS Workshop on Web Services and Agent-Based Engineering (2003) 57-66
8. Huhns, M. N.: Agents as Web Services. IEEE Internet Computing (July/August 2002) 93-95
9. Richards, D., Sabou, M., van, Splunter, S., Brazier, F.M.T.: Artificial Intelligence: a Promised Land for Web Services. In: Proceedings of The 8th Australian and New Zealand Intelligent Information Systems Conference (ANZIIS2003) Macquarie University, Sydney, Australia (December 2003) 205-210
10. Rao, S. A., Georgeff, M. P.: BDI Agents: From Theory to Practice. In: Conference Proceedings of 1st international conference on multiple agent system (1995) 312-319
11. Hendler, J.: Agents and the Semantic Web. IEEE Intelligent Systems (March/April 2001) 30-37
12. Li, Y., Ghenniwa, H. H., Shen, W.: Agent-based Web Services Framework and Development Environment. Journal of Computational Intelligence (2004)
13. Joseph, B., Kopena, William, C. Regli: DAMLJessKB: A Tool For Reasoning With The Semantic Web. In: 2nd International Semantic Web Conference (ISWC2003), Sanibel Island, Florida, USA (October 20-23 2003)
14. Mandell, D. J., McIlraith, S.: The Bottom-Up Approach to Web Service Interoperation. International Semant Web Conference (2003) 227-241