ASAPM – An Agent-based Framework for Adaptive Management of Composite Service Lifecycle

Mohan Baruwal Chhetri, Ingo Mueller, Suk Keong Goh, Ryszard Kowalczyk
Centre for Information Technology Research, Faculty of Information and Communication Technologies, Swinburne University of Technology, Australia
{mchhetri, imueller, sgoh, rkowalczyk}@ict.swin.edu.au

Abstract

Agent technology is well positioned to address some of the key problems of Service-oriented Computing. It can work together with other technologies such as Web/Grid Services, Semantic Web, Process technologies, and Component Software to contribute to wide adoption of the Service-oriented Computing paradigm. This paper presents a generic agent-based framework which can be used as reference architecture for developing agent-based systems for composite service life-cycle management. The focus is on providing techniques and tools for the adaptive service agreement and process management in order to ensure collective functionality, end-to-end QoS, and adaptive provision of complex services

1. Introduction

Significant research efforts are being directed into exploring the potential of the adoption of SOA using Web services and Semantic Web services technologies. The aim is to create new flexible approaches for programming distributed software systems that provide support for interoperability and integration to meet changing needs of modern inter-enterprise and cross-organizational collaborative business processes. Fundamental building blocks of SOA are service entities which are loosely coupled self-contained and self-describing software components. Multiple 'atomic' services can be combined into composite services (or workflows). One main focus of research interests is on the investigation of novel techniques for dynamic and adaptive management of the composite service life-cycle; meaning the partial or full automation of any of the life-cycle's phases starting from service publication, service discovery, service selection and contracting, service monitoring, Service Level Agreement (SLA) management, and enactment. In this respect a key challenge is to develop appropriate infrastructure and mechanisms for automated decision making or decision support.

Agent technology seems to be well positioned to address the key challenge because it complements Semantic Web technologies and provides abilities of autonomous operations, learning, adaptation, interactions, and cooperation in the context of distributed systems of multiple independent decision makers. Software agents can be used for automating business processes, performing adaptive decision making, and supporting coordination of resources in both, collaborative and competitive environments.

This paper presents generic agent-based framework along with a set of generic interaction protocols that can be used as reference architecture for building agent-based systems for composite service life-cycle management. The agent-based framework is the result of research carried out in the Adaptive Service Agreement and Process Management (ASAPM) project [8] that focuses on developing novel agent-based techniques and software tools for adaptive service level agreement management in order to ensure collective functionality, end-to-end QoS, stateful coordination, and adaptive provision of composite services.

This paper is organized as follows. Section 2 discusses related work. Section 3 outlines the ASAPM composite service life-cycle. Section 4 defines the ASAPM reference architecture. Section 5 describes the generic subsystem architecture and Section 6 the generic agent architecture. Section 7 concludes this paper.

2. Related Work

Plenty of research outcomes have been produced addressing automated dynamic composite service life-cycle management using agent technology. Sycara et al

0-7695-3028-1/07 $25.00 © 2007 IEEE
DOI 10.1109/WI-IATW.2007.88

[6] [7] present agent-based approaches for automated dynamic service discovery, selection, and enactment based on semantically enriched UDDI registries and matchmakers in combination with sophisticated broker agents. Similarly, Matskin et al [3], Howard et al [2], and Blake [1] describe agent-based approaches combining middle agents and Semantic Web concepts for semantic service description, discovery, matchmaking, and composition of composite services. Savarimuthu et al [5] describe an agent-based service composition framework for static and dynamic service composition and enactment in the context of supply chain based workflows where dynamic service selection is achieved by modeling composite services with Colored Petri Nets. Although the approach supports monitoring of functional and non-functional parameters during service enactment it does not provide dynamic exception handling since error messages are reported to human users. Nolan et al [4] introduce an architectural framework to tackle the difficulties that arise when service requests cannot be perfectly matched to existing service descriptions. This is done by enriching service descriptions during service negotiation and selection phase with additional information regarding negotiated functional and non-functional parameters which is stored in an additional ontology document. With this additional information available, the enactment of a composite service becomes possible.

All approaches above focus on aspects of dynamic service composition only. In contrast our approach is intended to addresses both, service composition and service enactment. It provides both phases with adaptive techniques to allow runtime negotiation, selection, and composition of composite services as well as dynamic exception handling for exceptions such as service failure and SLA violation. Dynamic exception handling supports automated re-negotiation of existing SLA contracts or re-planning activities for the entire composite service workflow depending on error type and enactment status. Finally, we aim at providing a framework that can be easily adjusted and extended for the development of Service-oriented applications in any application domain. So far we have validated our framework with the implementation of four different usecase scenarios from the Internet service provisioning (ISP), telecommunications, multimedia, and logistics sector within the ASAPM project. In addition, the negotiation component of our framework has been successfully integrated into another service provisioning platform in a collaborative effort with the European Union funded Adaptive Service Grid (ASG) project.

3. ASAPM Composite Service Lifecycle

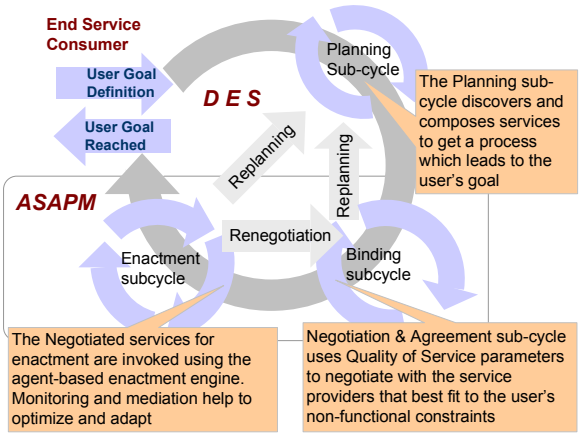

Figure 1. Role of ASAPM in Service Composition Lifecycle

The ASAPM framework has been modeled based upon the adaptive composite service lifecycle which consists of three sub-cycles – *planning*, *binding* and *enactment*. The planning subcycle is the first stage in processing a user request which is handled by the Domain Expert System (DES) as shown in Figure 1. It receives the user request which contains the functional and non-functional (QoS) requirements that need to be satisfied. In this phase, the DES composes an abstract composition that can satisfy the user request. It is assumed that each service within the composition can be fulfilled by several service providers. In the case of successful planning, the abstract composition is forwarded to the binding sub-cycle.

In the binding subcycle, abstract service compositions are transformed into executable service compositions by binding concrete service providers to each abstract service within the composition. This is a two step process. In the first step, the ASAPM platform tries to find the combination of services which best satisfy the user request. It involves the negotiation with several candidate providers over the provided QoS parameters. Once an agreement is reached with a particular service/s such that the end-to-end QoS requirements are met, the second step is the formation of contracts between ASAPM and the selected service provider/s. These two steps collectively form the binding subcycle. In the enactment subcycle the concrete service composition is enacted. The invocation of the services is monitored for two reasons – in order to ensure that the contracted QoS parameters are satisfied and also to create service profiles by aggregating the monitored data. This profiled data can

be used in the future by the Planning and Binding subcycles. At the end of the enactment subcycle, the user request is satisfied. Since web service environments are highly dynamic, it is necessary to handle service failures and QoS violations through adaptive mediation. Thus the enactment subcycle includes service enactment, service monitoring and mediation of service failures and QoS violations.

4. Overall ASAPM Architecture

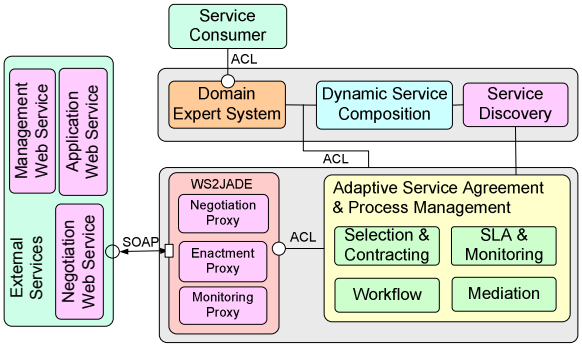

Figure 2. Overall ASAPM Architecture

Figure 2 shows the overall ASAPM architecture. DES handles the *planning* phase, Negotiation and SLA & Monitoring subsystems fulfill the binding subcycle. Similarly, Workflow, SLA & Monitoring and Mediation subsystems combine to fulfill the enactment subcycle. The ASAPM subsystems provide the following functionality:

1. *Negotiation subsystem* - is responsible for autonomous SLA negotiation and renegotiation of composite services in order to satisfy the end-to-end QoS of the service composition.
2. *SLA & Monitoring subsystem* - is responsible for SLA formation, management of SLA values during service enactment, and SLA termination. The monitoring component monitors the actual QoS during service enactment and triggers mediation if there is a QoS violation or service failure.
3. *Workflow subsystem* - is responsible for enactment of services according to the service composition definition. It updates the status of the process enactment as each atomic service is invoked successfully or otherwise.
4. *Mediation subsystem* - provides adaptive feature to the service composition life-cycle by providing mechanisms to handle the dynamics when it comes to handling service failures and QoS violations.

The ASAPM framework and reference architecture aims to serve as a middleware for service-oriented environments. It is built on top of already available standards such as SOAP and WSDL, and WS-BPEL. Domain specific ontologies can used in order to adapt the ASAPM Framework towards specific application domains. All interaction between ASAPM and the external web services takes places through SOAP over HTTP. Similarly all internal interactions within ASAPM take place using ACL. Interactions between the DES and ASAPM also take place using ACL. We have built the JADE adaptor to enable communication between non-agent based systems and multi-agent systems.

5. ASAPM Subsystem Architecture

Management of web service compositions involves two aspects – coordination of the entire service composition and management of individual services at the atomic level. The global coordination layer relates to management at the composition application level under global QoS constraints. It ensures collective functionality, end-to-end QoS and stateful coordination of the complex service. The atomic management layer relates to management of atomic services within the composition under local QoS constraints. It ensures individual functionality and local QoS whilst impacting global QoS constraints. Hence, the ASAPM framework adopts a two layered architecture as shown in Figure 3. The two-layered architecture applies to the following subsystems:

1. *Negotiation subsystem* - the Negotiation Coordinator Agent coordinates negotiation of the service composition so that e-2-e QoS requirements are satisfied. The Atomic Negotiator Agents (ANAs) conduct negotiations with the candidate service providers and try to achieve the best negotiation outcome for the provided local QoS constraints.
2. *SLA & Monitoring subsystem* - the SLA Coordinator Agent monitors the global QoS during lifetime of the service composition. The atomic Monitoring Agents monitor the atomic services for local QoS violation. Whenever a local QoS violation is detected, they notify the SLA Coordinator Agent which takes appropriate action by triggering the mediation subsystem.
3. *Workflow subsystem* - the Workflow Coordinator Agent setups of atomic enactment agents for decentralized/distributed enactment of the service composition. It also collaborates with the SLA Coordinator Agent and the Mediation Agent to ensure adaptive handling of QoS violation and service failures.

However, the *mediation subsystem* consists of a single Mediation Agent which mediates to handle global QoS violation.

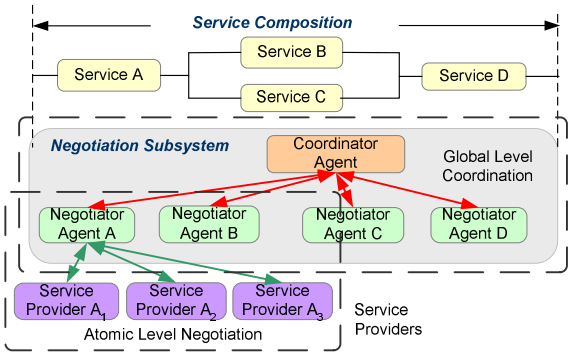

Figure 3. Two layer Subsystem Architecture

6. Agent Architecture

This section describes the different components which collectively make up the ASAPM agent as shown in Figure 4.

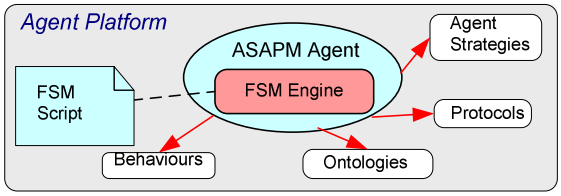

Figure 4. ASAPM Agent Architecture

6.1. Agent Behaviours

Every agent has a set of roles it has to play to accomplish a set of tasks. These roles are modeled as agent behaviours. We have defined a library of composable, reusable abstract behaviours which facilitates agent construction. These behaviours are extensible depending upon the application domain and also the user request. Table 1 shows the minimum set of behaviours we have defined for the Negotiation Coordinator Agent and the atomic Negotiator Agents.

6.2. Agent Communication Ontology

The ASAPM agents use Agent Communication Language (ACL) to communicate with one another. Each ACL Message is assigned a FIPA performative which indicates the type of message. Adherence to the FIPA Message protocol allows agents to distinguish between requests and responses. However nothing is known about the actual content of the messages. When agents within a system interact, they exchange information. This information may contain requests to agents to perform specific actions (AgentActions) which when executed, normally produce an effect and can generate a result which is then sent back to the requestor. We have developed a simple communication ontology based upon the FIPA Ontology Service specification. Some of the agent actions and corresponding concepts used by the agents within the negotiation subsystem are shown in Table 2.

6.3. Decision Making Strategies

Agents are flexible problem solvers which make context-aware decisions. The ranges of decisions depend upon the sophistication of the decision making strategies. We have developed decision making strategies for each subsystem – for coordinated-negotiation, QoS monitoring and the adaptive mediation of QoS violation and service failure. While making decisions, all ASAPM agents make use of persistent data which is stored internally.

Table 1. Abstract Behaviours (Negotiation subsystem)

Common Behaviours	AbstractListenBehaviour AbstractNegotiationCancelledBehaviour AbstractNegotiationFailedBehaviour AbstractNegotiationCompletedBehaviour
Negotiation Coordinator Agent Behaviours	HandleRequestBeahviour AbstractSetupNegotiationBehaviour QosDistributionBehaviour QosAggregationBehaviour ReceiveAtomicResultsBehaviour ReceiveConfirmBehaviour SendCompositeResultBehaviour ContractServicesBehaviour
Atomic Negotiator Agent Behaviours	ReceiveRequestBehaviour SendCFPBehaviour ReceiveCFPReplyBehaviour AnalyseOffersBehaviours ReceiveConfirmBehaviour ContractProviderBehaviour ReportResultBehaviour

For the negotiation subsystem one set includes strategies for the QoS distribution, QoS aggregation and the overall coordination of the negotiation [9] and is used by the NCA of the negotiation subsystem. Similarly, the second set of strategies support decision-making in atomic one-to-many negotiations [10] which are conducted between the ANAs and the service providers. For the SLA & Monitoring subsystem, we have monitoring rules and strategies defined at the global level and the atomic level. The monitoring rules at the global level determine whether the global QoS

has been violated or not and whether there is a need for mediation or not. The atomic monitoring rules determine whether the local QoS has been violated or not. We have also defined mediation strategies which determine how to handle QoS violation and/or service failures. These strategies are used by the Mediation Agent of the Mediation subsystem.

Table 2. Negotiation Ontology

Ontology Name: asapm.negotiation-ontology
Concepts: • NegotiateComposition-Result • NegotiateService-Result • Negotiation-Object • Negotiation-Offer
Agent Actions: • NegotiateComposition-Request • RenegotiateComposition-Request • NegotiateService-Request • RenegotiateService-Request • CancelNegotiation-Request • ResumeNegotiation-Request

7. Conclusion

This paper presents a generic agent-based framework which can be used for developing agent-based systems for composite service life-cycle management. It has been developed on the foundation of the adaptive composite service life-cycle and directly supports the binding subcycle and the enactment subcycle. It provides reference architecture for a multi-agent system which can be mapped to any application domain. In particular, it supports the automated service selection and composition of composite services based on QoS negotiations and SLA formation. In addition, it also supports dynamic exception handling for exceptions such as service failure and SLA violation through adaptive mediation. The reference architecture has been validated in four different business scenarios from the internet services provisioning, telecommunications, multimedia, and logistics application domains. In its current form, the framework handles the three subcycles in a sequential manner. Another approach is to interleave the different subcycles according to the workflow pattern of the composite service. We would like to explore this approach in future.

8. References

[1] Blake, B., *Coordinating Multiple Agents for Workflow-oriented Process Orchestration,* Journal of Information Systems and E-Business Management, Vol. 1, No. 4, 2003, pp.387-404

[2] Howard, R., and Kerschberg, L., *Brokering Semantic Web Services via Intelligent Middleware Agents within a Knowledge-Based Framework*, Proc. of the Intelligent Agent Technology, IEEE/WIC/ACM International Conference on (IAT'04), Beijing, China, 2004, pp. 513-516

[3] Matskin, M., Kunga, P., Rao, J., Sampson, J., and Petersen, S. A., *Enabling Web Service Composition with Software Agents*, Proc. of the Ninth IASTED International Conference on Internet and Multimedia Systems and Applications (IMSA'05), Honululu, USA, 2005, pp. 93-98

[4] Nolan, M., and Redpath, R., *Semantice Service Dissemination Architecture*, Proc. of the Twelfth Australasian World Wide Web Conference (AusWEB'06), Noosa Lake, Australia, 2006

[5] Savarimuthu, B.T.R., Purvis, M., and Purvis M., *Agent Based Web Service Composition in the Context of a Supply-Chain based Workflow*. Proc. of the First International Workshop on Coordination of Inter-Organizational Workflow: Agent and Semantic Web based Modules (CIOW'06), Hakodate, Japan, 2006

[6] Sycara, K., Paoulucci, M., Ankolekar, A., and Srinivasan, N., *Automated discovery, interaction and, composition of Semantic Web services*. Journal on Web Semantics, Vol. 1, No. 1, 2003, pp. 27-46

[7] Sycara, K., Paoulucci, M., Soudry, J., and Srinivasan, N., *Dynamic Discovery and Coordination of Agent-Based Semantic Web Services*. IEEE Internet Computing, Vol. 8, No. 3, 2004, pp. 66-73.

[8] Wu, B., Zhang, J. Y., Chhetri, M.B., Lin, J, Goh, S.K., et. al., *Adaptive Service Agreement and Process Management,* IEEE International Conference on Web Services (ICWS'06), Chicago, USA, 2006

[9] Chhetri, M. B, Goh, S., Lin, J., Brzotowski, J., and Kowalczyk, R., (2007), *Agent-based Negotiation of Service Level Agreements for Web Service Compositions*, Proc. of the Joint Conference of the INFORMS Section on Group Decision and Negotiation (GDN '07), 14-17 May 2007, Montreal, Canada

[10] Chhetri, M. B, Zhang, J. Y., Brzotowski, J., Goh, S., Lin, J., Wu, B., and Kowalczyk, R., (2006), *Experimentation with Three Different Approaches of Agent-based Negotiation*, Proc of the Workshop on Service-Oriented Computing and Agent-based Engineering (SOCABE '06) held in conjunction with the Fifth Joint International Conference on Autonomous Agents & Multi-Agent Systems (AAMAS 2006), 8-12 May 2006, Hakodate, Japan