AN EVALUATION OF SERVICE FRAMEWORKS FOR THE MANAGEMENT OF SERVICE ECOSYSTEMS

Ravi Khadka, Department of Information and Computing Sciences, Utrecht University, The Netherlands, ravi@cs.uu.nl

Amir Saeidi, Department of Information and Computing Sciences, Utrecht University, The Netherlands, amir@cs.uu.nl

Slinger Jansen, Department of Information and Computing Sciences, Utrecht University, The Netherlands, slinger@cs.uu.nl

Jurriaan Hage, Department of Information and Computing Sciences, Utrecht University, The Netherlands, jur@cs.uu.nl

Remko Helms, Department of Information and Computing Sciences, Utrecht University, The Netherlands, remko@cs.uu.nl

Abstract

A service ecosystem is a marketplace for trading services in which services are developed, published, sold and used. Service ecosystems have changed the way of service delivery and service consumption among actors/parties, who perform specific roles for the operation of the ecosystems. Such actors, being service providers, consumers, mediators and intermediaries, ensure the livelihood of the ecosystem. However, the role of the service infrastructure provider, one of the actors of the service ecosystem, is still not being explored sufficiently. The service infrastructure provider provides service infrastructures/frameworks upon which other actors of the service ecosystem operate. In this paper, an evaluation framework for the service framework is defined, which is based on the features that are required for a service ecosystem to thrive. The evaluation framework is used to evaluate three open-source service frameworks. The evaluation framework facilities the selection process of a service framework among the largely available ones.

Keywords: SOA, Service Ecosystem, Service Frameworks, Apaches Axis2 Web Service Framework, Petals Web Service Framework, WSO2 Web Service Framework

1 INTRODUCTION

Recently, business has changed rapidly with the advancement in Internet technologies, enabling cross-enterprise collaboration. Web Service technology that follows the Service-Oriented Architecture (SOA) (Erl 2005) based architectural style has contributed substantially to the efficiency of cross-enterprise collaboration and to address the accelerating pace of changes, such as intra-organizational changes, changes in market demands, rules and regulations, and the changes in the supporting technologies. Lately, the successes of the companies like Amazon, Google, and Salesforce, have demonstrated the real commercial success of web service technology as a means to create value for customers. Such commercial successes have given rise to web service ecosystem that has radically changed the way we discover and invoke service in the Internet. A web service ecosystem is a logical collection of web services in which service delivery are subjected to constraints at business level (Barros & Dumas 2006). The service ecosystem is an evolutionary step of SOA in which services are not only means to achieve heterogeneous application integration but services are envisioned as tradable products. As a result, web service ecosystems are the marketplaces for trading services in the Internet.

While trading services, service ecosystems bring together the consumers and service providers in which various other actors from different legal bodies are involved for supply, delivery, and consumption of services. As envisioned by Barros & Dumas (2006), service ecosystems consist of the following actors: *consumers*, *providers*, *brokers*, *mediators*, and *intermediaries*. However, Barros & Dumas (2006) fail to mention an essential actor within a service ecosystem, the service infrastructure provider, which provides the infrastructure/framework upon which the other actors of the service ecosystem operate. The service infrastructure provider is an enabler for proper functioning of other actors in the service ecosystems as it provides computing infrastructure and/or set of additional functionalities such as service cataloguing (i.e., registry), composition, monitoring, and versioning of services in the service ecosystems.

In this paper, we define an evaluation framework based on the features of the service framework. The term features refer to the functionality (to-be) provided by the service framework of the service ecosystem. Those features are identified from the phases of the service life-cycle and service provisioning. The service infrastructure provides various functionalities to support service deployment, publication & discovery, composition, monitoring, negotiations, etc., at different phases of the service life-cycle and service provisioning (Cardoso 2008). Service life-cycle encompasses a variety of functions to support a service in all its phases from development to retirement (Raisanen 2005). Service provisioning is the act of preparing the system for the use of a service by a consumer, which facilitates the usage of the service in both technical and business aspects (Polan 2002). Recently, a significant number of service frameworks have become available in the market, which makes it difficult for the service infrastructure providers to choose the appropriate one. This has motivated us to develop an evaluation framework that helps the service infrastructure providers to evaluate and choose the appropriate framework among the largely available ones. Based on the evaluation framework, we present an evaluation of three open-source web service frameworks by investigating whether the frameworks offer the functionalities specified as evaluation criteria. To the best of our knowledge such an evaluation framework and the evaluation of the service frameworks are not yet available.

The rest of the paper is structured as follows: Section 2 presents the related work in the service ecosystem, Section 3 elaborates our evaluation framework, Section 4 introduces three open-source service frameworks, Section 5 summarizes the results of the evaluation, and finally, Section 6 presents our conclusion and identifies topics for future work.

2 RELATED WORK

Lately service ecosystems have received significant interest (Barros 2006; Riedl 2009). A service ecosystem is a logical collection of services in which several parties/actors ensure the livelihood of the ecosystem. Barros et al. identified the following actors (Figure 1) in a service ecosystem. *Service providers* offer services at the first place in terms of implementation and publish the service description. *Service consumers* discover and use those published services. *Service brokers* enhance service delivery by bringing the service providers and consumers together. The service brokers might also integrate a service with certain delivery functions such as payments and authentication or perform service composition. *Service mediators* generate value by customizing the provider's standard services towards consumer's need by translating between different service formats. *Service intermediaries* facilitate the service ecosystem by offering service delivery components (like payment, authentication, monitoring) that a service provider can use to create marketable services. Thus, through the flow of services from providers, mediators, and brokers a service is finally offered to a consumer. This flow leads to a separation of service delivery and consumption in service ecosystems.

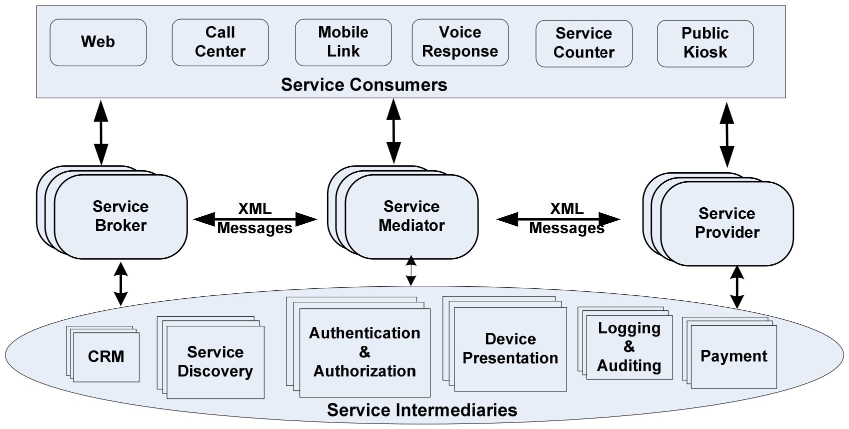

Figure 1. Top-level Architecture of a Service Ecosystem (Barros & Dumas 2006).

Another actor, which does not receive sufficient attention from service ecosystem research, is the service infrastructure provider that provides service infrastructure/framework (Riedl 2009). Such an infrastructure provides overall platform and complex functionalities upon which other actors of service ecosystems operate. The services provided by the service infrastructure bring together a large number of consumers, brokers, mediators, and providers and enable the business interaction among them. *Service providers* offer their services by publishing service descriptions in registries that are provided by the infrastructure. *Service brokers* customize the basic services according to consumers' need. *Service consumers* discover published services by querying the registries and select services from *service providers/brokers* based on the functionality, pricing, and quality of service, availability, or rating. The consumer will have to pay a due amount for the service consumption to provider. In this process of service supply and consumption, the service infrastructure serves the service ecosystem by providing functionalities such as service publication through registries, deployment, discovery, execution, composition, monitoring, security, compensation. With all these functionalities, the service framework already has potential role towards the success of service ecosystem (Riedl 2009).

Further, in the work of Cardoso et al. (2008) and Scheithauer et al. (2009), the roles of the service ecosystems are described. Riedl et al. (2009) has addressed the importance of service infrastructure provider in the service ecosystem and Riedl et al. (2009^{a}) presents the quality aspects to be addressed in the service ecosystems.

The evaluation of service infrastructure frameworks for service ecosystem is motivated by the ServiciFi (ServiciFi 2010) project that aims to extract services from monolithic products and open source components in the financial domain using Service Extraction Process[1] (SEP). The SEP consists

[1] For details, we refer to http://servicifi.files.wordpress.com/2010/06/serviceextraction.pdf

of the following steps: pattern identification and mining, service identification, source code extraction, service annotation, service compilation and assembly, service deployment, and service publication. After extraction, the extracted services need to be deployed in a service framework. However, the number of such service frameworks available in the market has increased substantially, so choosing between them has become a difficult task. This has motivated us to develop an evaluation framework in order to facilitate the evaluation and selection of the appropriate service framework from the largely available ones. We use the features present in the service infrastructure as evaluation criteria and such evaluation criteria are based on the requirements of the ServiciFi project. So, we do not claim the completeness of the evaluation framework. However, we believe that we have included the important features as evaluation criteria in our evaluation framework such that one can use this framework to select appropriate service framework.

3 EVALUATION FRAMEWORK

We defined our evaluation framework based on the features present in the service framework at different phases of service life-cycle and service provisioning. Those functionalities are identified from the relevant literature for service life-cycle phases (Wall 2006; Raisanen 2005; Pessoa et al. 2008; McBridge et al. 2007) and service provisioning (Polan 2002; Raisanen 2005). Typically, a service life-cycle is divided into design-time and run-time aspects. The design-time aspect includes activities for service development such as identification, modelling, and creation of services. However, in our evaluation framework we did not include the design-time aspect because we focus on evaluating the technical functionalities of the service framework after the services are deployed. The overall evaluation framework is depicted in Figure 2. Based on run-time life-cycle aspects, we have the following evaluation criteria:

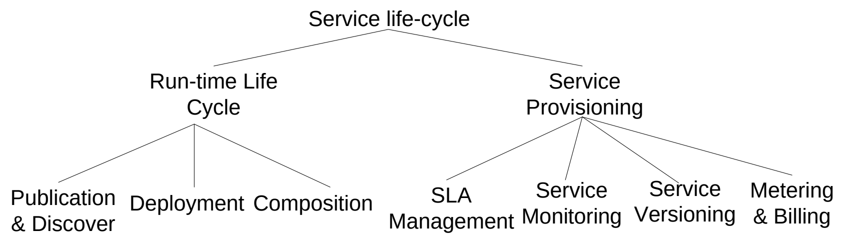

Figure 2. Evaluation criteria of the evaluation framework.

Publication & Discovery: The publication feature allows the service provider to advertise services using service description documents (e.g. WSDL, OWL-S, and SAWSDL). While discovering services, such published services are searched by consumers based on their requirements. Usually, the service description documents are published in a registry, where consumers can query to find the appropriate services. In order to facilitate service publication & discovery, a service framework can have a registry component. Under this criterion, we investigate whether the service framework has registry, which web service standards are supported for publication & discovery, and whether there is support for semantic annotations.

Deployment: This feature enables installing, configuring and managing service instances in the service framework. In deployment phase, the service framework can facilitate automatic code generation (skeleton and stubs) to ease development, hot update and hot deployment of services to manage run-time service installation. Under this criterion, we investigate whether the service framework facilitates code generations, hot deployment, and hot update of services.

Composition: This feature facilitates the aggregation of basic and/or existing services to new composite service. Service composition is one of the central concepts of service ecosystem so under this criterion, we investigate whether the service framework has integrated orchestration engine and if BPEL standard is supported.

Service provisioning includes activities that are required to manage and control the behaviour of services during usage. A service framework needs to provide functionalities to facilitate service provisioning. Our evaluation framework includes the following criteria:

Service monitoring: The separation in service supply and consumption in a service ecosystem can alter the initial service properties offered by the provider. Such properties could be functional (i.e., KPI for management reporting and optimization of services) and/or non-functional (i.e., cost, trust, security, availability, reliability, etc). The brokers and service intermediaries augment services offered by the providers before it is delivered to the consumers, hence it is likely that the initial service properties would change. A service monitoring feature within a service framework can be very useful. Such monitoring feature creates trustworthy environment among the actors. Under this criterion, we investigate whether the framework supports QoS monitoring, run-time monitoring of services, and business process monitoring.

Service Versioning: Due to changing functional and non-functional requirements to align with the business requirements, services need to evolve and this leads to multiple versions of original services (Fang 2007; Bechara 2008). Thus, service versioning is inevitable in the service ecosystems. Under this criterion, we investigate whether the service framework supports the web service versioning.

SLA Management: SLA is a negotiated contract between service provider and consumer that define the term under which the service need to be provided. Prior to subscribing a service, the consumer and the service provider negotiate the terms in which rights as well as obligations of both parties are described. Therefore, SLA management is an important aspect, particularly in a multi-party service delivery scenario as in service ecosystem (Riedl et al. 2009).

Metering & Billing: The subscription of a service by a consumer has to be recorded as per usage and the billing information has to be generated for payment. Thus, it is desirable to have such facility in a service framework.

Based on aforementioned evaluation criteria, we evaluate three open-source service frameworks in the following section.

4 WEB SERVICE FRAMEWORKS

In this section, we present brief overview of three open-sources Web Service Frameworks (WSF) namely: Apache Axis2, Petals WSF, and WSO2 WSF/C++. We explain each WSF on the basis of their high-level architecture and briefly explain the main modules, which provide more information about the functionality present in the WSF. Such an explanation of the high level architecture also enhances the evaluation. For instance, function of any specific module in the architecture gives information about the functionality provided by the WSF such as, presence of code generation module facilitates the automatic generation of skeleton from the service description files.

4.1 Apache Axis2 Web Service Framework

Apache Axis2 is an open-source web service framework developed by Apache Software Foundation (Axis2 2010). Apache Axis2 provides an object model and a modular architecture to add functionality and support for new web services-related specifications and recommendations. For the evaluation purpose we used Apache Axis2/Java 1.5.4. Figure 3 shows the key components in Axis2 architecture.

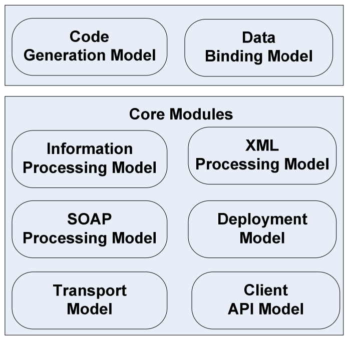

Figure 3. *Apache Axis 2 WSF.*

XML Processing Model encapsulates the complexities of XML processing by using AXIs Object Model (AXIOM), which is an XML infoSet-based representation of SOAP message. *SOAP Processing Model* involves the processing of incoming SOAP messages, which is divided into various phases that the execution of processing will walk though. *Information Processing Model* manages service lifecycle and service session of static and dynamic state of service. *Deployment Model* allows deployment of the services, configuration of transports, and extension of SOAP processing model. The deployment mechanism includes hot deployment, and hot updates of services. *Transport Model* allows to use and to expose same service in multiple transports that are supported like HTTP, SMTP, JMX, and TCP etc. *Client API model* provides a convenient API to interact with services using Axis2. *Code Generation Model* is used to generate server-side (skeleton) and client-side (stub) code also with descriptors. Data Binding Model enables Axis2 to encapsulate the XML-infoset to process the SOAP and also provides programming language-specific interface.

Apache Axis2 does not have service registries, which limits the capability of service publication and discovery. Apache Axis2 supports WSDL1.1 and WSDL2.0 but does not support semantic annotations. While deploying the web services, both the contract first approach (i.e., code generation from service description files like WSDL) and code first approach (i.e., generating service description files from source code) are supported. Also, to support run-time installations and updates of services, hot deployment and hot update of services are supported by Axis2. The lack of integration with service composition engine limits Axis2 from supporting service composition functionality. Apache Axis2 does not have any service monitoring functionalities. However, an extension for QoS monitoring has been reported for Apache Axis2 by Ma et al. (2005). There is no support for service versioning, SLA management and metering & billing.

4.2 Petals Web Service Framework

Petals WSF is an open-source web service framework developed by petals Link (Petals 2010). The petals service framework incorporates followings: *Petals ESB*, *Petals studio*, *Petals master*, and *Petals view*. The Petals WSF is depicted in Figure 4.

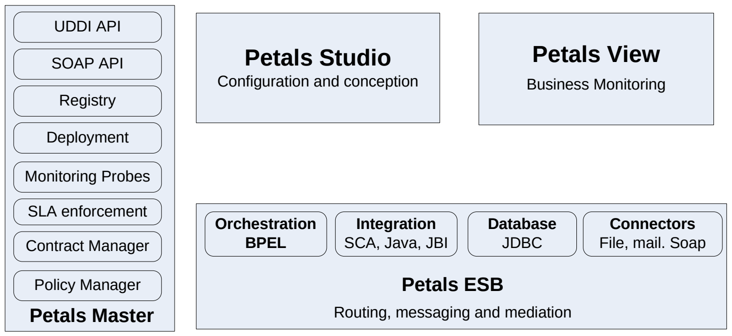

Figure 4. *Petals WSF .*

Petals ESB is an open-source ESB for routing and messaging in SOA and real-time deployment (hot deployment) of services for large scale enterprise SOA solution. The key features of Petals ESB include routing, orchestration and integration, and hot service and node-server deployment. *Petals studio* is an Eclipse-based tool providing support for the configuration, design, and development of SCA components and BPEL processes facilitating the development phase of services. *Petals master* is a SOA governance solution for service administration that allows an enterprise to organize, enforce, and reconfigure the service resources. The key features of petals masters include service indexing and documentation, registry for publish & discovery, contract management through SLA, and service versioning. *Petals view* monitors the data exchanges between services, creates key performance indicators for management reporting, and optimization of services and business processes.

Petals WSF facilitates the service publication and discovery through the service registry available in *Petal masters* that supports WSDL 1.1 and WSDL 2.0 standard. The availability of UDDI API and its integration with Eclipse enable to query UDDI registry for discovery. Petals WSF is also featured with EasyWSDL, a WSDL parsing library. EasySAWSDL, an extension of EasyWSDL, also allows developers to provide semantic annotations in service description. While deploying the services, the code generation functionality, hot installation and hot updates of services are supported. Petals WSF has an integrated process orchestration engine for service composition that allows creating designing, validating, and executing BPEL-based process. Petals WSF provide service monitoring facilities for service administration using the monitors available in Petals master and Petals view components. The *Petals master* supports the service management, service run-time management and SLA management. The *Petals view* provides real-time monitoring and analysis of business processes. Although the documentation of the *Petals master* claims the support for service versioning, there was no any support in practice. There is no support for metering & billing in Petals WSF.

4.3 WSO2 Web Service Framework

WSO2 WSF is an open-source service framework developed by WSO2 (WSO2 2010). WSO2 provides three variations of the framework for C, C++, and PHP. For our evaluation, we used the framework for C++ (WSO2/C++), which provides APIs to implement web services and web service clients. The WSO2 WSF/C++ is depicted in Figure 5.

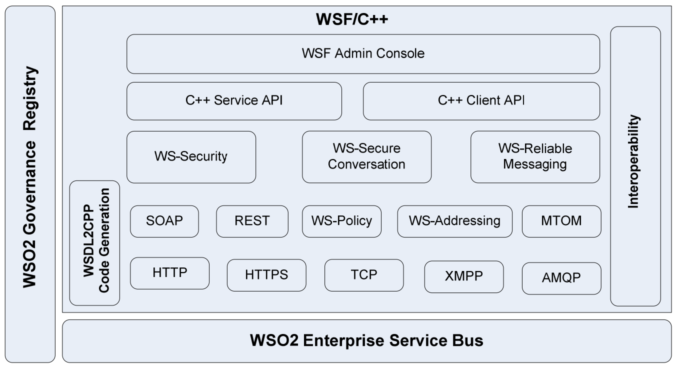

Figure 5. WSO2 WSF.

WSF/C++ provides APIs that help to build SOA applications. The WSF/C++ is a viable solution for system integrations because WSF/C++ facilitates legacy C++ applications to be exposed as web services. *WSO2 governance registry* is an SOA governance solution for service administration. The registry enables users to associate lifecycles with the specific services and also provide a customizable enterprise security model. *WSO2 ESB* is an open-source ESB with enhanced management, development support, and SOA governance capabilities. The distinguishing features of WSO2 ESB include the possibility to create proxy services, which are the virtual services hosted by the ESB, and to provide QoS support to such proxy services. *WSO2 ESB* supports hot deployment and hot update of services.

WSO2/C++ has a registry service that allows service providers to publish their services in WSDL 1.1 and WSDL 2.0 standard and discover those services. However, semantic annotations are not possible in WSO2/C++ framework. The code generation is supported. WSO2/C++ supports hot update and hot installation of services. Apart from run time monitoring of services, other functionalities under service provisioning such as QoS monitoring, process monitoring, service versioning, SLA management, and metering & billing are not present in the WSO2/C++ framework.

5 DISCUSSION

Table 1 summarizes the results of the evaluation of the service framework described in Section 4, according to the criteria defined within our evaluation framework.

Publication & Discovery feature determine where the services are stored, how effectively are the services described, and how can the services be searched. Usually the service registries are used to store the services for publication and discovery. Further, service publication and discovery are based on keyword searches in which the service provider nominates the keywords. Such publication and discovery can be fostered by adhering WSDL standard in which message input, output, and operation details are factored into keywords. In all the evaluated frameworks, there is support for WSDL for service description. Furthermore, semantic annotation of WSDL, which can enable a higher level of automation in service discovery, negotiation, and composition (Maleshkova 2009), is not yet supported in most of the frameworks. Service registry, which not only enhances the service publication & discovery, but also facilitates the governance aspects, is present in two of the evaluated frameworks.

Evaluation Criteria			Apache Axis2 WSF	Petals WSF	WSO2 WSF
Life-cycle Phases	Publication & Discovery	Registry	o	●	●
		WSDL 1.1	●	●	●
		WSDL 2.0	●	●	●
		Semantic Annotation	o	●	o
	Deployment	Code Generation	●	●	●
		Hot Update	●	●	●
		Hot Installation	●	●	●
	Composition	Orchestration Engine	o	●	o
		BPEL Support	o	●	o
Service Provisioning	Service Monitoring	QoS Monitoring	Θ	o	o
		Run-time Monitoring	o	●	●
		Process Monitoring	o	●	o
	Service Versioning		o	o	o
	SLA Management		o	●	o
	Metering & Billing		o	o	o

o: Not Supported ●: Supported Θ: Third party support

Table 1. Summary of the evaluation

Deployment facilitates installing, configuring, and managing the service instances. A code generation facility allows the developers to automatically generate the skeleton code and thereby assists in development. In a service ecosystem, services evolve due to change in requirements and it is desirable to have run-time installation and updates of such changes. The concept of hot deployment and hot update in web service is the solution for run-time installation of service changes. Hot deployment and hot update refer to the ability of making changes to a running instance without causing any downtime. In all the web service frameworks we have evaluated, code generation, hot deployment, and hot updates of services are supported.

Composition is the central concept of service ecosystems (Barros 2006) in which the existing services, provided by service providers, are aggregated to a new composite service and delivered to service consumers. Service composition not only realizes a composite service, but also accelerates the reuse of existing services (Khadka & Sapkota 2010). To perform service composition, a service framework can integrate orchestration engine that supports BPEL, an OASIS standard executable language for specifying service composition. We believe the presence of such orchestration engines can significantly increase the usability.

Service Monitoring facilitates trust among the various actors within a service ecosystem. The separation in service supply and consumption can alter the initial service properties offered by the provider, especially when the original services are augmented by the brokers and intermediaries. Service monitoring functionalities such as QoS monitoring and process monitoring are lacking in the evaluated frameworks. But, run time monitoring is available in Petals and WSO2 WSF.

Service Versioning is an important aspect as services evolve continuously to address the changing business requirements and, thereby, leads to multiple versions of the original services. In this dynamic scenario, service versioning becomes a priority to minimize the impact of changes made to the service. So far, none of the evaluated service frameworks support service versioning.

SLA Management encompasses the SLA contracts that include precise definitions of service reliability and responsive guarantees, and penalties that apply when these guarantees are breached between service provider and service consumer (Barros 2006). The aspects of SLA include non functional properties like pricing, availability, trust, reputations, penalties, etc. In particular, SLA management becomes an important aspect in a multi-party service delivery scenario as in service ecosystem. The Petals WSF is the only framework to provide support for SLA management in our evaluation.

Metering & Billing are specific functionalities for payment as per service usages by the service consumer. It might be the case that the consumer pays to service intermediaries that augment the services provided by the service providers or directly pays to the service provider. In both cases, the metering and billing of the service usages are the basis for payment. In all the evaluated service frameworks, metering & billing facilities are not present.

So far, we have evaluated three open-source WSFs with respective to their core functionalities, which foster the management of the service ecosystems. From our evaluation we observed that most of the functionalities within the phases of service life-cycle are present in the WSFs. Typically, the current standards of web service technology are being adhered. For instance, publication & discovery phases support WSDL standard. The functionalities required for service development in the deployment phase are supported. The integration of orchestration engine to support service composition is still not mature. However, the crucial functionalities like QoS monitoring and service visioning of service provisioning are yet not supported.

6 CONCLUSION

In this paper, we have presented an evaluation framework, based on features present in the service frameworks for the management of service ecosystems. The features were identified at the different phases of service life-cycle and service provisioning. We initially discussed the service infrastructure provider as an enabler of the service ecosystem. The main objective of service infrastructure provider is to provide the ecosystem with a service framework within which other actors operate. The functionalities provided by such service frameworks largely determine the success of service ecosystems. The evaluation framework was used to evaluate the three open-source web service frameworks. The evaluation framework was based on the requirements of our ServiciFi project, so we do not claim the completeness of the evaluation framework. However, we believe that we have included the relevant features such that the service infrastructure providers can use this evaluation framework to select an appropriate one from the available ones in the market. Based on our evaluation, we found Petals WSF as a promising service framework for our ServiciFi project despite the fact that service versioning, metering & billing criteria are not available. Our choice of Petals WSF does not reflect the fact that Petals WSF is the best open-source service framework available. The choice of service framework largely depends on the specific requirements of the service infrastructure provider, in our case the alignment with the ServiciFi project goals.

There are several limitations in our evaluation: we only selected open-source WSFs for evaluation but there is large number of proprietary service frameworks available in the market. In future work, we extend our evaluation by including such proprietary frameworks. The evaluation framework can be further enhanced by including various other functionality criteria such as type of composition supported (i.e., static vs. dynamic composition support), verification of compositional correctness support, discovery of services based on non-functional qualities like pricing, quality of service, user rating. Also, the inclusion of performance of the frameworks as an evaluation criterion is an issue to be considered as future work.

References

Axis2. (2010). Apache Axis2: an open source service framework. Available from: http://axis.apache.org/axis2/java/core/

Barros, A. and Dumas, M. (2006).The rise of web service ecosystem. IT Professional, 8 (5), 31-37.

Bechara, G. (2008). *Web service versioning.* Available from: http://www.oracle.com/technetwork/articles/web-services-versioning-094384.html

Cardoso, J. and Voigt., K. and Winkler, M. (2008). Service engineering for the internet of services. In Proceedings of the 10th International Conference on Enterprise Information Systems (ICEIS2008), p. 15–27, Barcelona, Spain, Springer.

Erl, T. (2005). Service-oriented architecture: concepts, technology, and design. Prentice Hall PTR.

Fang, R. and Lam, L. and Fong, L. and Frank, D. and Vignola, C. and Chen, Y. and Du, N. (2007). A version-aware approach for web service directory. In Proceedings of IEEE International Conference on Web Services, p. 406-413, Utah, USA, IEEE.

Khadka, R. and Sapkota, B. (2010). An evaluation of dynamic web service composition approaches. In Proceedings of the 4th International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing, ACT4SOC 2010. p. 67-79, Athens, Greece, INSTICC press.

Khadka, R. and Sapkota, B. and Pires, L. and van Sinderen, M. and Jansen, S. (2011). Model-Driven Development of Service Compositions for Enterprise Interoperability. In Proceedings of the 3rd International IFIP Working Conference on Enterprise Interoperability (IWEI 2011). p. 177–190, Stockholm, Sweden, Springer .

Ma, W. and Tosic, V. and Esfandiari, B. and Pagurek, B. (2005). Extending apache axis for monitoring of web service offerings. In Proceedings of the IEEE International Workshop on Business Services Networks, p. 44 - 51, Hong Kong, China, IEEE.

Maleshkova, M. and Kopecky, J. and Pedrinaci, C. (2009). Adapting SAWSDL for semantic annotations of restful services. In: On the Move to Meaningful Internet Systems: OTM 2009 Workshops, p. 917–926, Springer.

McBride, G. and Fritz, P. and Rialho, B. (2007). Delivering SOA solutions: service lifecycle management. Available from: http://www.w3.org/TR/wslc/

Pessoa, R.M. and Silva, E. and van Sinderen, M. and Quartel, D.A.C. and Pires, L.F. (2008). Enterprise interoperability with SOA: a survey of service composition approaches. In *Proceedings of Workshop on Enterprise Interoperability*. Munich, Germany.

Petals. (2010). Petals: An open source SOA integration solution. Available from: http://www.petalslink.com/en}{http://www.petalslink.com/en

Polan, M. (2002). *Web service provisioning*. IBM Developerworks. Available from: http://www.ibm.com/developerworks/library/ws-wsht

Raisanen, V. and Kellerer, W. and Holtta, P. and Hikkinen, S. (2005). Service management Evolution. In *Proceeding of IST Mobile and Wireless Communications Summit,* p. 67. Dresden, Germany.

Riedl, C. and Bohmann, T. and Leimeister, J. M. and and Kremar, H. (2009 [a]). Quality management in service ecosystem. *Information Systems and E-Business Management,* 7 (2), 199-22.

Rield, C. and Bohmaan, T. and Leimeister, J. and Kremaar, H. (2009). A framework from analysing service ecosystem capabilities to innovate. In *Proceedings of the 17th European Conference on Information Systems*, Verona, Italy, Available at SSRN: http://ssrn.com/abstract=1650662

Scheithauer, G. and Augustin, S. and Wirtz, G. (2009). Describing Services for Service Ecosystems. In: *Service-Oriented Computing–ICSOC 2008 Workshops*, p. 242–255, Springer

Wall, Q. (2006). Understanding the service lifecycle within a SOA: run time. Available from: http://www.oracle.com/technetwork/articles/entarch/soa-service-lifecycle-run-099156.html

WSO2. (2010). WSO2: Web services framework. Available from: http://wso2.com/products/web-services-framework/