Solution Marketplace for Service Composition and Integration

Hamid R. Motahari-Nezhad, Jun Li, Bryan Stephenson, Sven Graupner, Sharad Singhal

HP Laboratories
HPL-2009-96

Keyword(s):
Cloud services, Service Composition, Service Integration, Service Adaptation, Solution Marketplace

Abstract:

Service composition and integration are well investigated problems in SOA. However, they still remain among the hard SOA challenges for which automated approaches have yet to be developed. These issues are hindering the agile and cost-effective development of service-based business solutions, and the need for addressing them becomes more pressing with the increase in the number of online (cloud) services. We refer to a composition of services that solves a business problem as a (composition) solution. In this position paper, we argue that solution reuse at a large scale can be exploited to address challenges of service composition and integration by harnessing the collective intelligence and labor of various businesses and people present on the Internet. We propose a reference architecture and technical design of a platform for representation, sharing, and search of solutions and also a marketplace which foster the reuse of service composition and integration solutions.

External Posting Date: May 6, 2009 [Fulltext] Approved for External Publication
Internal Posting Date: May 6, 2009 [Fulltext]

To be published in 3rd International Workshop on Web Service Composition and Adaptation, Los Angeles, USA July 2009

Solution Marketplace for Service Composition and Integration

Hamid R. Motahari-Nezhad, Jun Li, Bryan Stephenson, Sven Graupner, Sharad Singhal
HP Labs, Palo Alto, CA, USA
{hamid.motahari,jun.li,bryan.stephenson,sven.graupner,sharad.singhal}@hp.com

***Abstract*—**Service composition and integration are well investigated problems in SOA. However, they still remain among the hard SOA challenges for which automated approaches have yet to be developed. These issues are hindering the agile and cost-effective development of service-based business solutions, and the need for addressing them becomes more pressing with the increase in the number of online (cloud) services. We refer to a composition of services that solves a business problem as a (composition) solution. In this position paper, we argue that solution reuse at a large scale can be exploited to address challenges of service composition and integration by harnessing the collective intelligence and labor of various businesses and people present on the Internet. We propose a reference architecture and technical design of a platform for representation, sharing, and search of solutions and also a marketplace which fosters the reuse of service composition and integration solutions.

Keywords-Cloud Services; Web Service Composition; Web Service Integration; Web Service Adaptation;

I. INTRODUCTION

The number of services offered on the Internet is growing fast pushing us a step further to realizing the vision of offering everything-as-a-service [1]. This is fostered by the opportunities created by technological advances of Web services, REST-based services, Web 2.0, mashups and recently cloud computing. Cloud computing has provided an unprecedented opportunity for an economical and large scale offering of software and hardware resources as services [2]. Along with this trend of growth in the number of online services, we are witnessing the following two important phenomena:

First, services that are available on the Internet are not all Web services (with WSDL interfaces). Many of them are only advertised on the Web targeting people and using Web forms and other means such as email and FTP for the exchange of input and output. Many services are also offered following a REST-based architecture, i.e., their functionality and usage methods are only described in documentation. The variety of service types makes the automation of service integration and composition tremendously difficult.

Second, web services are traditionally developed and used in enterprise settings to address enterprise-scale integration and process management requirements. However, users and providers of online services now include small and medium businesses (SMBs) as well as individuals. This implies that millions of online users and thousands of SMBs are becoming service providers and users, a phenomenon we refer to as *service clouds*. Similarly, services that are offered by large businesses in the cloud environment are also consumed by small businesses and individuals.

Therefore, there will be thousands of online services, most of which are targeting or will be used by the long tail of users, i.e., small businesses and individuals. This adds up to millions of service users. Examples of cloud service providers that are targeting this market segment are Amazon and Salesforce.com, where users can develop and deploy new services. The technology is now ever closer to enabling the formation of virtual businesses [2], businesses in which most or all functions are outsourced to online services.

In this paper, we focus on the issues of integration and composition of cloud services into solutions for small businesses and individuals that are using them for a business purpose. The key observation in this setting is that the number of business scenarios in which online services need to be composed and integrated is far fewer than the number of possible combinations of such services. This is because many of the possible service combinations do not make sense or they are not interesting from a business perspective. Furthermore, unlike the enterprise computing environment, in which integration and composition solutions have to adopt the complex business processes in the environment, SMBs are more willing to accept the business processes and the business applications that are commonly defined in their domains, with little or no modification to meet their own needs. Therefore, the needs for integration and composition of services into solutions built to address similar business problems are recurring.

By *solution* we mean a reusable composition of services solving a business problem. It is assumed that the business problem is of a more complex nature than is currently addressed by a single service. Therefore, a number of services need to be *composed* and *integrated* to form a solution. A solution may need to be complemented with functionality for which no external service has been found and hence needs to be developed. At the end, the user of a solution sees the aggregate as a whole, not the individual parts, solving his or her business problem.

Creating solutions requires effort. In the traditional enterprise, system integrators design and build software solutions comprised of standardized software building blocks such as SAP and integrate them into solutions that are ready to use by a line of business. Most of the composition and integration is a manual, time-intensive and costly effort, an approach that does not meet the speed, agility, and cost requirements of SMBs. Creating lighter-weight solutions much faster and more efficiently is a challenge.

Devising automated approaches to address these problems considering different types of services, which we discussed earlier, is still among the hard challenges of SOA [3]. While there are semi-automatic approaches that facilitate service composition and integration under certain assumptions [3], these approaches still require manual intervention with labor-intensive tasks by domain and IT experts in producing a workable solution.

Given the recurring nature of such tasks across similar business needs of SMBs, we believe that solution sharing and reuse provide the foundation to greatly simplify the service integration and composition problems, particularly for the SMB, and can meet the needs of the majority of customers within the same business domain.

Reuse has been explored and practiced in different settings including software practice over the past fifty years [4] and enterprise settings. However, the opportunities for solution reuse among SMBs are much greater than for process and software reuse in enterprise settings, due to increased customization requirements in those settings.

In this paper, we propose the design and development of a platform for sharing and reusing solutions for cloud services. Unlike existing software and process reuse work, where templates or patterns have been defined as reusable artifacts, service integration and composition solutions are the building blocks. In summary, the contribution of this paper is threefold:

(i) As service composition and integration involves considerable manual effort, and many of these needs are recurring, we propose harnessing the collective intelligence and labor available in the cloud service environment.

(ii) We propose a reference architecture for a platform which enables the sharing and reuse of composition and integration solutions.

(iii) One of the inhibitors to reusing software artifacts is the lack of incentives. We discuss various issues and aspects including business models that stimulate solution reuse in such a platform and provide insights into offering a marketplace for composition and integration solutions.

The rest of paper is structured as follows. In Section 2, we present the background on solution reuse. In Section 3, we present the opportunities and challenges of a platform and marketplace for service integration and composition solution reuse. Section 4 presents the proposed reference architecture for such a platform, and Section 5 discusses approaches to enabling a solution marketplace around this platform. We present related work in Section 6 and conclude the paper in Section 7.

II. BACKGROUND

The idea of reuse is not new and it is well-investigated in various contexts including software reuse, component reuse, and more recently process reuse, which are discussed in the following.

A. Software Reuse

Different software artifacts have been the subject of reuse including code, modules, components, frameworks, architectures, templates and patterns [4]. Code reuse refers to using existing code, as a function, subroutine or module, in developing a new application and in more than one place. The related concept of software component reuse was introduced in 1968 by Douglas McIlroy. Software libraries aggregate a set of reusable components addressing recurring problems (such as converting one format to another) with well-defined inputs and outputs. In order to foster component and software reuse, usually software repositories are created in enterprises with guidelines from software product line engineering. There exists also work on how to search software repositories to automatically retrieve software components, e.g., based on the signature of input and output [5].

Design patterns [6] offer a disciplined approach to capture and document recurring problems and their solutions, however, at a different level of abstraction than code. Each design pattern includes a description of a problem, and associated template of a solution typically demonstrated through an example. For a given problem at hand, the proposed solution template has to be instantiated.

B. Process Reuse

The notion of reuse has been also explored in the area of business processes (see Figure 1). Workflow patterns [7] are reusable control flow constructs that can be used in the design of processes. Each pattern comes with a description, example usage, and how it is supported in various workflow products. The MIT process repository (process.mit.edu) offers reusable and high level descriptions of processes in various domains. Process templates

present a generally accepted procedure for performing particular business activities in a given domain. Examples of process templates are RosettaNet PIPs, which offer templates for processes such as placing an order or its payment in the context of supply chain. Another example is business blueprints from SAP. These blueprints need to be customized by experts considering requirements of a particular business.

Process reuse refers to the reuse of a concrete definition of a process without the need for extensive customization. Some process templates in a commercial product, e.g., from SAP, come with solutions that can be claimed to practice this form of reuse. However, the solution reuse in such settings is limited due to the need for manual, often extensive customization. Furthermore, within the boundary of an enterprise, the opportunities for reuse of a process are also limited as often the same process is executed in one place of the enterprise. However, within the SMB environment, these limitations do not exist. Therefore, there are better opportunities for solution reuse across small businesses.

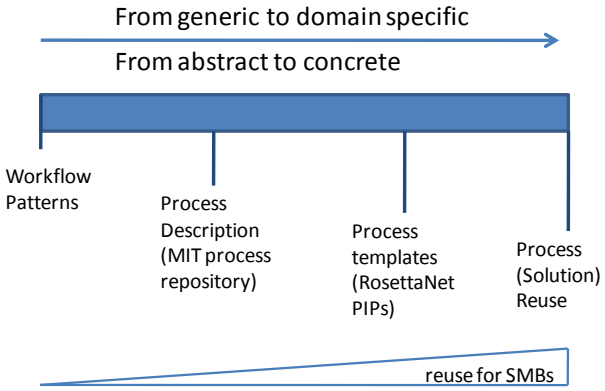

Figure 1. The spectrum of artifact reuse in business processes.

C. Community-driven Application Development

Open source project repositories such as SourceForge[1] offer environments for sharing and reusing software artifacts. Software enhancements are contributed back to the software developer community.

Recently, technologies related to Web 2.0 have provided an unprecedented opportunity for non-technical users to reuse and contribute in application development. In particular, developing and sharing mashup applications (e.g., using Yahoo! Pipes and Microsoft Popfly) [8] demonstrate the enormous potential of leveraging community efforts in developing new applications by reusing existing software and services. For instance, programmableweb.com enables listing and sharing mashup and light-weight web applications. It maintains references to more than 5,000 mashups and Web APIs, provided by the community. Currently, it reports a very steady increase in the number of applications and APIs that are registered over the last 6 months.

These examples show that if an enabling infrastructure is available the power of the crowd can drive the creation of novel and diverse artifacts.

III. SOLUTION REUSE FOR SERVICE COMPOSITION AND INTEGRATION

To use services in various business scenarios, users (either SMBs or individuals) need to compose them. The need for composition also implies the need for integration of services to enable them to interact seamlessly.

The service composition and integration problems have been widely investigated for enterprise settings [3] and also for individual user settings, most recently through mashup applications [8]. The recent focus in enterprise settings has been on the integration and composition of applications rendered as WSDL-based Web services. With mashups, the focus has been integration and composition of REST-based interactions and user interface-driven services [8]. Despite the fact that various semi-automated and software engineering–based (e.g., pattern-based [4][6]) approaches have been investigated for service composition, these problems are still recognized as hard challenges of SOA [3].

In this paper, we propose a platform for sharing and reusing "solutions" of composition and integration of services. We view a solution as an integration and composition of concrete artifacts capable of being deployed and executed. A solution includes concrete binding definition for the specific set of services that are integrated or composed. A solution is offered (shared) by a *solution provider* and used by a *solution user*. A solution in this context

[1] sourceforge.net

may be delivered as a piece of code (e.g., BPEL or Java code) or service package that can be readily deployed (e.g., as virtual machine image), or may be offered as a service hosted by the solution provider.

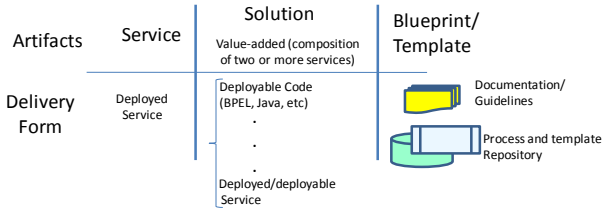

Figure 2. The contrast of "solution" vs service, and blueprints/templates.

Figure 2 illustrates the relationship between a solution, a service, blueprints that are often delivered in the form of documentation or guidelines, and also solution templates that are instantiated for a given problem.

A. Why and When Reuse Solutions?

The first question which needs to be answered about solution reuse is how to understand when solution reuse is suitable and for what situations. Indeed, the effort to find, understand and use a solution should be considerably less than the effort to develop the solution from scratch. Furthermore, if the solution that is considered for reuse requires more than 20-25 percent change and customization, then it is known that it is more efficient and effective to rewrite the solution rather than reuse it [9].

Considering the business needs of SMBs, most of them prefer applications to *configure* (change only settings) rather than having to *customize* (write additional software to fit their needs) partially due to cost and lack of expertise. On the other hand, there are excellent opportunities for solution reuse among SMBs because often they have similar business needs and require little or no customization. This is also witnessed by the number of SMBs that are customers of SaaS (Software as a Service) providers such as Salesforce.com compared to the portion of SMBs running customized in-house solutions [10]. SMBs can benefit more than large enterprises from solution reuse, and prefer solution reuse to templates and patterns, as they are looking for ready-to-use artifacts.

B. Challenges of a Solution Reuse Platform

From the technical perspective, there are many aspects that must be considered when creating a platform for sharing and reusing solutions. We broadly present them in the following categories:

Representation of solutions: The integration and composition solution may be provided as simply as a reference to the resultant composite service (SaaS), or as complexly as application packages that contain BPEL, Java programs, and configurations. Common abstractions are needed to represent solutions to simplify reuse. The abstractions should include meta-data about the solution, e.g., which services are integrated/composed, which versions, supported business function definitions and applicable business domains, etc. Furthermore, each solution should also come with a set of test cases to allow solution integrators to verify solution correctness.

Organization and Discovery of solutions: Solutions need to be organized in a way that makes it easy for users to find the solution candidates through searching, catalog browsing, and referrals. Instead of a fixed browsing scheme, solutions may be structured in a multi-modal manner, such as a concept hierarchy of business domains or mind maps. A large solution repository becomes cluttered and makes solutions hard to browse and find. Methods for search and retrieval of solutions are required to address (i) how to express user needs, e.g., to enable expressions based on the existing solutions in the repository? The level of knowledge of users must also be considered. (ii) How to find the most relevant solutions and provide a ranked list of solutions? And if the exact matching solution does not exist, how to find a set of solutions that are available and can be potentially composed further to achieve the functionality that the customer needs?

Quality of solutions: One issue that has been hindering reuse in software libraries is the poor quality of shared solutions [4]. The approach for software libraries has been to appoint inspectors to ensure the quality of contributed materials before posting them to the library. In the community-driven portals, it has been shown that the feedback from the community can serve an important role in identifying the quality through a rating system. Enabling the solution reuse community to rate composed solutions can provide valuable information on solution quality without requiring extensive testing. Sharing of test-related assets, such as test cases or even test suites, among different solution customers may improve the solution quality.

Granularity of solutions: One important issue that affects the reuse is the granularity of solutions. If a solution includes too many services, it may apply only to a small number of business needs. On the other hand, the issue with

solutions composing too few services is that they have to be composed together to fulfill a business need. Enabling configurability of solutions is an approach allowing automatically generating a configured solution (in different granularities) based on customer needs.

Enabling solution reuse at a large scale*:* Solution reuse is for both solution users and providers so that they can build on top of existing solutions. If creation of further solution variants requires a comparable amount of effort as that spent on the first solution, solution reuse cannot reach the economies of scale and drive down the cost of the development. There is a need for a runtime infrastructure support, toolset and methodologies for rapid solution reuse.

C. Challenges of a Solution Reuse Marketplace

The success of solution reuse not only requires addressing the technical challenges of creating a platform, but also a business environment that provides incentives for both the solution provider as well as the solution user. We refer to it as a *marketplace*, which is built on top of and around the platform. Awareness of legal and business regulations is important for the marketplace.

Drawing from the early success of the cloud service providers such as Amazon.com and Salesforce.com, we envision a marketplace in which solution providers and solution users can engage for mutual benefit. Unlike the open source community, which only imposes rules (through licensing schemes such as GPL), there are various issues that the marketplace needs to take care of in order to make solution reuse a viable option for SMBs. This is to ensure that SMBs can leverage online services with significantly reduced cost and acceptable risk. To achieve such objectives, the following are the main concerns that need to be addressed.

Solution SLA: Defining quality attributes and ensuring quality of solutions expressed as a service level agreement (SLA) is very important for a business to use a solution. The SLA essentially is a legal basis under which solution provider and solution user operate. If a solution is provided by the composition of a set of services, it should be clear how SLA and quality criteria are guaranteed given that there may not be agreements between solution providers and service providers. In particular, a method is needed to detect and handle SLA violations.

Traditional code reuse is based on complete ownership by the code provider. In the context of solution reuse, the responsibilities and liabilities need to be clarified and agreed upon. For instance, if the solution does not meet expected SLA, it may not be due to the solution itself, but due to one of the underlying services that have been composed.

Insurance: The marketplace or a third party service provider can be an insurance issuer. By providing certification mechanisms and financial backup in case of violations, when the integrated solution does not work (due to any reason) and the customer's business is disrupted, the customer's loss can be covered by the insurance policies purchased.

Pricing models: In the marketplace both free and commercial solutions may be offered. The pricing model for commercial solutions ranges from buying a solution license (such as normal software), paying usage fees (if offered as a service), or the solution may be provided free of charge but the solution provider makes money through revenue sharing with underlying services.

Business Needs Driving Reuse: Whether a solution is reusable mostly depends on whether there is a common business need for it. Also, identifying the right granularity of composition solutions may also depend on business needs. Therefore, providing an environment and mechanisms for customers to communicate business needs to solution providers is valuable, especially in the context of SMBs.

IV. PLATFORM REFERENCE ARCHITECTURE

In this section, we propose a reference architecture for a platform for sharing and reusing solutions addressing the challenges identified in Section 3.

To support the search, configuration, sharing and integration of a business solution, we envision there is a collaborative environment between solution users, solution providers, and solution integrators (solution integrators help SMBs to integrate solutions into their existing business service suite).

The proposed architecture is presented in Figure 3. It has three layers: collaboration portal, services and business solution repository. We will illustrate the details of the platform using an example use case.

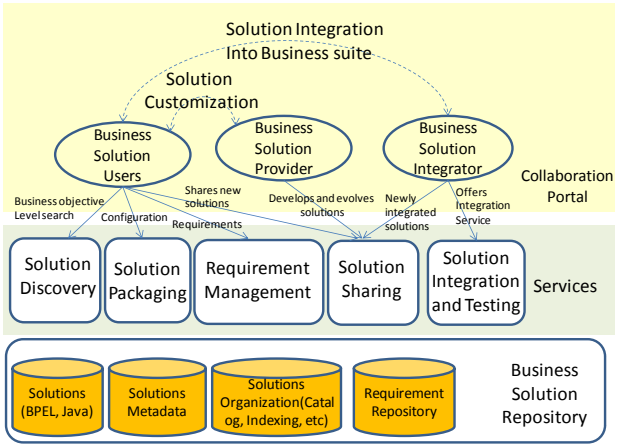

Figure 3. A Collaborative Environment to Support Business Solution Reuse.

A. Examle Use Case on Reuse

Gwen runs a small local insurance agency and is interested to take advantage of the many available services to automate some of her insurance agency tasks. A simple example is the task of scheduling an appointment with a customer where she manages her customers through a customer relationship management (CRM) service to which she has subscribed. This task requires a calendar service capable of holding and managing appointments, email and/or phone services to set the appointment with customers and CRM to update the scheduled appointment time in the CRM service once it is settled. Aside from appointment scheduling, customer interaction history can be automatically tracked through the CRM service. She may not find one service providing all the above capabilities (note that this example is purposely simple) and therefore she needs a solution putting all of these services together. This business need is not unique to Gwen and many other agencies require a similar solution.

If a platform for sharing solutions is provided, a solution provider may fulfill the above need by composing the CRM service (e.g., from Salesforce.com) with calendar, email and phone services from other service providers. Through browsing the platform, Gwen may select among business solutions that have already been built and tested by solution providers. The chosen business solution can then be integrated, e.g., by the solution integrator contracted by the solution provider, into the agency's business service suite. A deployable business solution from the solution provider may consist of an executable business process definition (such as defined in BPEL) for the task, a set of services that are involved in this business process and the configuration parameters for the business process definition and the chosen services.

B. Solution Sharing and Discovery

Business solution sharing and discovery allow solution providers to publish business solutions and solution users to find business solutions that satisfy their needs.

Each business solution is represented by two sets of information: (i) the solution package, e.g., a Java code library, BPEL code, or service endpoints, and configuration specification and supporting materials (test cases, and Web content if the solution is packaged as a Web application), and (ii) description of the solution including the business domain, pricing model, documentation, and details of services that are composed.

Each business solution will be categorized under a business classification that follows the common terminology defined by the marketplace. For example, the customer appointment scheduling will be under: "Customer relationship management → appointment scheduling". The solution will also list the services that it relies upon. For each type of service in the solution (e.g., the calendar service), the solution provider identifies the recommended and compatible services that have been tested (e.g., Google Calendar and Zoho Calendar). The required information about a solution is a natural outcome of the solution development processes and thus does not impose extra burden to the solution providers.

The collaboration portal enables users to search the repository based on business objectives. The business objectives are formulated in terms of the description of existing solutions including which services have been composed. The search criteria also include factors such as solution popularity (how many customers are using the solution), customer ratings and service charges. In addition, the marketplace should enable advanced search for technical users to express their requirements on the specifications of solutions. For example, a BPEL definition associated with the solution should be searchable. The search function may take into account data dependencies

between the services. Such dependencies are provided by the solution users. For example, to specify the desired properties for the customer appointment solution, the query may require the "existence" of the following services "CRM service, calendar service and email service". In addition, the user may specify that calendar service should precede email service. BPEL-based search techniques like the one presented in [11] are helpful to enable the search of the repository based on high level description of desired composition logic, as well.

C. Business Solution Packaging

When the business solution is retrieved from the solution repository, it has different options and feature sets. In the customer appointment example, the solution provides the following configuration options: with or without the CRM service; setting appointments with email, with phone or both. The packaging service helps the customer determine which option to choose based on capturing information (e.g., through a set of questions or probing the customer environment) on the current services that are already deployed in the solution user's business environment or based on the popular configurations selected by other customers.

Once a particular business solution configuration is chosen, and the involved services are identified, the solution packager generates the specific business solution package corresponding to what the solution user specified in the configuration phase. An interesting problem to pursue is the automatic generation of the solution from configuration data.

D. Requirement Management Service

The solution discovery service may find no existing solutions that meet the user's exact requirements. For example, the insurance agency may feel that to better communicate with agents on their mobile phones, the calendar service needs to be integrated with a phone service and accept appointment confirmations from a phone. The collaboration portal enables the solution user to communicate to the solution provider about (i) their specific solution requirements, (ii) the solution in the repository which is closest to their needs, and (iii) the preferred service providers.

The solution provider receives the customization requirements and modifies the existing solution accordingly. Later, if a customized solution becomes popular, then the solution provider can turn it into a regular solution offering.

E. Solution Integration and Testing Service

The integrator is responsible to take the solution the user has chosen from the solution provider, and integrate it into the user's business environment. The solution integration occurs at three layers: business logic, data, and presentation (user interface) [12].

A challenge of such integration is how to test the solution without disturbing the running business of the solution user. Due to the typical small per-instance solution scale in the SMB domain, one approach is to simulate or clone its entire business suite into an isolated environment in which the solution is integrated and tested. Once the solution is tested, the solution can be transferred into the running business environment.

V. SOLUTION MARKETPLACE

We envision the solution reuse platform will be part of a marketplace which facilitates interactions and commercial exchange between solution providers and users as marketplace participants. The marketplace will create an efficient market for reusable business solutions. Marketplace participants may have a legal contract with the marketplace provider rather than with each service provider. This arrangement avoids contracts between every pair of participants exchanging value in the marketplace.

The marketplace will track information about services to help customers understand and manage the risks involved in doing business in this manner. This includes the history of feedback from customers, monitoring service quality, various certifications such as SAS-70 [13] and new certifications for this purpose, and third-party information such as information from the Better Business Bureau[2]. In this way, a growing body of evidence will be collected which provides a level of confidence that solutions will meet expectations.

Historical information may not be sufficient for a solution user to engage with a set of services. In many cases, SLAs will outline specific requirements for technical and legal terms. Services will arise in the marketplace which monitor, validate, and insure SLAs. A key capability of the marketplace is to resolve disputes that will inevitably arise between marketplace participants. The marketplace will be in a position to vet participants and guarantee payment for services, which will facilitate doing business with unknown companies.

The record of service metrics maintained by the marketplace will be a key enabler for the creation of insurance coverage against business disruption or losses due to term violations. Once enough information exists to quantify

[2] www.bbb.org

risks to an adequate level of confidence, insurers will get into this market. Periodic audits, certifications, and other assurance methods will be required of an insured solution provider. Adjusting the insurance premiums based on provider performance and quality will create an incentive for providers to improve their operations.

To ensure the solution user's continuous operations, the marketplace can find and provision a second source (backup) service when a primary service is unavailable. The marketplace can enforce contractual and technical requirements which ensure that a solution user's data can be retrieved from services using a data migration service.

VI. Related Work

Existing work for service composition that exploits the notion of reuse can be divided into two categories:

(i) providing higher-level abstractions, e.g., as a service component [14] that encapsulates a composition solution in the form of a reusable and extendible package, and

(ii) pattern-based composition [15], in which a set of control flow and workflow patterns as well as application specific patterns are used to establish the composition. Nevertheless, no prior work exists for providing a platform for reusing service composition and integration solutions.

Zlatkin and Kaschek [16] propose a repository of processes in order to foster process reuse and provide a conceptual architecture for it. They suggest presenting processes in the CoCA process modeling language and to provide an SQL-like language for retrieving processes. In this paper, we present a platform and a marketplace enabling solution reuse for service integration and composition. We envision the offering of common abstractions (metadata) for solutions rather than representing solutions using specific languages.

The idea of reuse is also explored in the context of data integration. For instance, [17] presents an approach to schema matching by learning from a corpus of previous matches. However, the issue of how the information-base is built and managed is not explored. The proposed platform and marketplace in our work offer technology and business foundations which are useful in realizing the complementary vision of "service parks" [18]. Service parks are envisioned to be formed by communities of related services.

Commercial cloud service providers such as Boomi[3], Bungee Connect[4] and Cast Iron[5] provide integration solutions at the technical level. These provide pre-built adapters for many enterprise applications, and also languages in which data and application integration logic can be developed by programmers. However, they do not support large scale sharing and reuse of integration solutions developed by the community. Salesforce.com, on the other hand, has offered AppExchange[6] that enables sharing add-ons that are developed on top of salesforce.com applications and capabilities. It provides a catalog listing of such applications. There are currently hundreds of applications in payroll management, service and support surveys, recruiting, etc., that have been built by partners and developers. The entire application package (which is either free or comes with its own pricing model) can be imported into a new development account, from which the imported application can be modified to fit the new needs. The solution provider might impose certain usage policies (such as $10/month/seat) to the solution users for using its shared application package. Currently, Amazon.com and Google also allow solution providers to provide their own service capabilities and bundle them into the customized solutions, e.g. as an Amazon Machine Image (AMI)[7] or as a Google App[8].

Similarly, Apple provides a software marketplace called AppStore[9] for applications for the iPhone. However, for now it is not possible to compose applications; they have to be bought and consumed independently.

VII. Conclusion

Some features and capabilities of the proposed platform for service integration and composition solutions are implemented to various degrees at existing cloud platform and service providers. However, the application reuse at these platforms, e.g. salesforce.com, is limited to the add-ons built on top of each provider's solutions. Also, the proprietary solution stack is fully controlled by the platform provider. On the other hand, other platform providers support reuse for enterprise-level applications.

We anticipate that there will be communities of service providers that jointly provide services that can be composed and integrated, specifically in the context of solutions targeting SMBs. We envision a solution marketplace to allow solution providers and solution users to share business solutions and have such business

---

[3] www.boomi.com
[4] www.bungeeconnect.com
[5] www.castiron.com
[6] www.salesforce.com/appexchange
[7] developer.amazonwebservices.com
[8] code.google.com/appengine
[9] www.apple.com/iphone/appstore

solutions integrated into the user's business environment by providing the collaboration support environment. We believe that the proposed architecture and technical design for the platform will play an important role in enabling creation of a community-driven and provider-supported solution marketplace.

The presented approach and the platform are also very useful in the context of application services on mobile devices, where there is a need for application service composition. The presented platform in this paper enables building and sharing composite application services from thousands of existing mobile services for personal and business purposes.

REFERENCES

[1] S. Robison. The next wave: Everything as a service. Executive Viewpoint, 2008. http://www.hp.com/hpinfo/execteam/articles/robison/08eaas.html.

[2] H.R. Motahari-Nezhad, B. Stephenson, S. Singhal, "Outsourcing Business to Cloud Computing Services: Opportunities and Challenges," Technical Report HPL-2009-23. January 2009.

[3] M.P. Papazoglou, P. Traverso, S. Dustdar, F. Leymann, "Service-Oriented Computing: State of the Art and Research Challenges," IEEE Computer 40(11): 38-45, 2007.

[4] I. Jacobson, M. Griss, P. Jonsson, Software Reuse: Architecture, Process and Organization for Business Success, Addison-Wesley, 1997.

[5] A. M. Zaremski and J. M. Wing. Signature Matching: "A Tool for Using Software Libraries," ACM Trans. Softw. Eng. Methodol., 4(2):146-170, 1995.

[6] Erich Gamma , Richard Helm, Ralph Johnson, John Vlissides, Design patterns: elements of reusable object-oriented software, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1995.

[7] W.M.P van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros, "Workflow Patterns," *Distributed and Parallel Databases*, 14(3), pages 5-51, July 2003.

[8] D. Benslimane, S. Dustdar, A.P. Sheth, "Services Mashups: The New Generation of Web Applications," IEEE Internet Computing 12(5): 13-15, 2008.

[9] R. L. Glass, Facts and Fallacies of Software Engineering, Addison-Wesley, 2003.

[10] Jeffrey Kaplan , How SMBs Can Save Money Using SaaS, February 2009. http://itmanagement.earthweb.com/entdev/article.php/3803136/How-SMBs-Can-Save-Money-Using-SaaS.htm.

[11] C. Beeri, A. Eyal, S. Kamenkovich, and T. Milo, "Querying business processes," In Proceedings of the 32nd international Conference on Very Large Data Bases, Seoul, Korea, September 12 - 15, 2006.

[12] G. Hohpe, and B. Woolf, Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions, Addison-Wesley, 2003.

[13] Statement on Auditing Standards No. 70 (SAS 70), American Institute of Certified Public Accountants (AICPA) http://www.aicpa.org/download/members/div/auditstd/AU-00324.PDF.

[14] Jian Yang, Mike P. Papazoglou, "Service components for managing the life-cycle of service compositions," Inf. Syst. 29(2): 97-125, 2004.

[15] Haitao Hu, Yanbo Han, Kui Huang, Gang Li, Zhuofeng Zhao, "A Pattern-Based Approach to Facilitating Service Composition," GCC Workshops 2004: 90-98.

[16] Zlatkin, S. & Kaschek, R. "Towards Amplifying Business Process Reuse," ER 2005 Workshops, LNCS 3770 – 0364, Springer, 2005.

[17] J. Madhavan, P. A. Bernstein, A. Doan, and A. Halevy, "Corpus-Based Schema Matching," In Proceedings of the 21st international Conference on Data Engineering, April 05 - 08, 2005.

[18] C. Petrie, and C. Bussler, "The Myth of Open Web Services: The Rise of the Service Parks," IEEE Internet Computing 12(3): 96-95, May 2008.