SOCA (2007) 1:93–95
DOI 10.1007/s11761-007-0011-2

INTRODUCTION

A view on service-oriented architecture

Jen-Yao Chung · Kuo-Ming Chao

Published online: 15 May 2007

The evolution of software has been accelerating this decade due to the rapid change of the global business environment. Business requires more agile and flexible software systems which can be easily and swiftly customised to meet its changing demands. One of the solutions for this challenge is to shorten the software development process by composing relevant and reliable software components or services to provide required application functionalities. Meanwhile, an evolutionary computing paradigm named service-oriented architecture (SOA) is also emerging to provide architectural solution for business and it has attracted great attention from both industry and research communities due to its promise of achieving innovative business value by minimizing the gap between software systems developments and the way business operates. Since SOA is an emerging paradigm and it is still evolving, its nature and potential are not yet fully realised by research communities at present.

From a conceptualized perspective, a service is the non-material equivalent of tangible goods which, in this context, can be interpreted as a repeatable business task e.g. checking customer's credit, opening a new account, etc. Service-orientation is a way to integrate business as linked services which bring about the desired outcomes, and a service-oriented application is a set of orchestrated services that support a business process built on an SOA. Therefore, SOA can be viewed as an IT architectural style that supports service orientation. Based on defined and standardized interfaces, SOA provides flexible connections to allow integration among distributed and heterogeneous components and enable real-time monitor of them as well. As a result, users are able to access key business measures in a timely manner. Besides, SOA also empowers users to have sufficient control over business processes through event-driven interactions among users and services so that the users can receive information and respond to the system in time.

From a lifecycle perspective, SOA is an architecture supporting iterative development process where software systems are created through four stages including *model, assemble, deploy,* and *manage* [1]. In the *model* phase, business abstraction is carried out to reach a common understanding of the business processes, objectives and outcomes between business and IT. The applications, which are built upon the resulting model, need to meet the defined business requirements. Similar to the traditional software engineering method, the business abstraction in SOA starts by analysing business requirements from gathering and eliciting business process and related data. The resulting requirements and gathered information can be used to model and optimize the business processes after simulating them. Consequently, the model can be used as a benchmark to measure business performance. After the required business model has been established, the *assemble* phase takes place. Maximising reuse of the existing services, resources and applications to form part of the new system is one of the main tasks in this phase. Therefore, it starts by discovering existing services and applications which can contribute to the new system. However, the existing applications may not be compatible with SOA specifications so it is essential to transform them or wrap them up into services. In the case that there is no existing functionality which can satisfy the new business process requirement, creation of new services

J.-Y. Chung (✉)
IBM T. J. Watson Research Center
P.O. Box 218, Yorktown Heights
NY 10598, USA
e-mail: jychung@us.ibm.com

K.-M. Chao
Department of Computer and Network Systems
Coventry University, Coventry, UK
e-mail: k.chao@coventry.ac.uk

has to be carried out. Prior to their delivery, testing on these services must be conducted to ensure their reliability. Once the required services are available, the implementation of the required business process can be fulfilled by orchestrating the services and to be ready for deployment. Before the *deploy*ment of the business process, the process requires a robust, scalable, and secure services environment for execution. Hence, the runtime environment needs to be configured and scaled to a level that meets quality requirements of the business process. A services environment not only provides all the essentials to run mission-critical business processes reliably but it also provides the flexibility to make updates dynamically in response to changing business requirements. Once the service environment has been configured appropriately, it facilitates numerous point-to-point integrations to reduce maintenance cost and to manage system complexity. The objective of the *manage*ment phase is to maximise the awareness of the service environment to the developers and users by allowing them to perform real-time monitor of key performance indicators so that it is possible to make better business decisions sooner than previously. The information required for identifying the root causes of problems must be provided in order to resolve them and feed them back to the business process model and thus it can enable continuous improvement. In addition to managing underlying service assets, the *manage* phase includes the control of service version issues and the provision of good performance by improving service availability and response time.

In the light of aforementioned development process, an essential construct in SOA named enterprise service bus (ESB) [1] is required to facilitate the integration and choreography of diverse services and support event-based messaging among services as well. In comparison with monolithic enterprise application integration (EAI), ESB is more adaptable and flexible to ever-changing business environments. ESB is a reference architecture which can be implemented by middleware technologies to facilitate communication among different services and applications. ESB not only includes a collection of key services to assist the SOA developers in building and managing the services but it also encloses a set of services to support better decision-making with real-time information. Since the main objective of ESB is to support the development of SOA applications and to provide a flexible connectivity for services and applications, the components or services in the ESB can be varied according to the system or customer requirements. The following illustrates some services that the ESB could or should have. An ESB should contain a set of infrastructural services to support the other services in the architecture by optimising system throughput, availability and performance. *Business innovation* and *optimisation services* utilise the outputs given by the related services to produce just-in-time and relevant business information for the decision makers. In the ESB, it is essential to have an appropriate *IT service management* that can manage and secure services, applications, and resources. *Development services* provide an integrated environment for the design and creation of solution assets. It includes *interaction services* that enable collaboration between processes and information. Orchestrating and automating business processes are supported by *process services*. The *information service* manages diverse data in a unified manner to ensure that the information can be accessed consistently. *Partner services,* including essential business information, are used to maintain and connect with trading partners so the partners can be integrated within the SOA development. *Access services* facilitate interactions with existing information and application assets. With the above supporting services and architecture, *Business application services* can build upon a robust, scalable, and secure service environment. In principal, the ESB architecture is designed to define the IT services that are necessary to support SOA at each stage in the iterative development process. The advantage of the ESB is that with the support from industry commercial development tools, it allows application developers to focus on meeting the core business needs rather than the interoperability issue among varied IT infrastructures. In addition, the ESB can also easily accommodate new services as additions to the existing services or adapt to any changes in the services. As a result, SOA is not only a vehicle for an organisation to deliver system integration and add business value but it also is a platform to enable collaboration across organisations.

With these recent advancements in SOA methods and technologies and with successful stories of using SOA for business processes, the value of SOA is gradually being recognised academically and industrially. However, SOA is still at its infant stage as its benefits have not been fully explored and a number of issues have not been addressed. SOA bridges the gap between business and IT by widening collaboration and drawing together the expertise from business and IT. However, the mismatch between business requirements and IT specifications could still exist and narrowing this gap still has room to improve in order to automate the design process. Current SOA technologies focus on supporting service integration and collaboration in terms of interface and functionality but less attention has been paid to the issues of service content integration which could lead to service provision inconsistency. Besides, Using SOA has scale-emerging properties so service marketplaces for selling and buying services will be needed in order to form a sound economic sustainability. Pricing and delivery policies as well as licence and copyright issues in a service-oriented market also require further investigation. As services evolve, they become a product of a continuous co-creation process among service providers and consumers and the implication of the SOA approach for the change of business culture and society needs to be studied. To address these issues may

require inputs from researchers in the Social Sciences. The aforementioned issues are a minimal list, so we believe that Service and SOA can stand as a subject area on its own and its advancement requires the collaboration and contribution from various disciplines. Such a subject needs to address: the scientific aspects associated with service creation and innovation; the engineering aspects of constructing SOA applications by applying existing knowledge to generate the new values of services; and management aspects to improve the process of creating and capturing service values.

References

1. IBM's SOA Foundation (2005) An architectural introduction and overview. http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf