Towards Value-Driven Business Modelling Based on Service Brokerage

Yucong Duan[1], Keman Huang[2], Ajay Kattepur[3], and Wencai Du[1]

[1] College of Information Science and Technology, Hainan University, China
duanyucong@hotmail.com, wencai@hainu.edu.cn
[2] Department of Automation, Tsinghua University, China
victoryhkm@gmail.com
[3] ARLES Group, INRIA Paris-Rocquencourt, France
ajay.kattepur@inria.fr

Abstract. Service engineering is an emerging interdisciplinary subject which crosscuts business modeling, knowledge management and economic analysis. To satisfy service providers' profiting goals, the service system modeling needs to take care of both the short and long run customer satisfaction. The ideology of value driven design fits well for this need. We propose to work towards value driven design by introducing a form of service design patterns, we call *service value broker*(SVB), with the aim to shorten the distance between economical analysis and IT implementation and increase the value added on all sides. SVB allow us to not only study the value added in terms of functional and business aspects, but also reason about the need for brokerage across various domains. In this paper, we model the basis of SVB and its network based organization architecture in the background of Cloud.

1 Introduction

Software design patterns [1] have been well studied with formal semantics and applications in multiple domains. In case of service oriented computing (SOC) applications, such design patterns[2] may yield a standard for composing services dependent on improvement in functional, Quality of Service (QoS) or business contractual aspects. Most existing work focuses on a specific functionality or quality property from a technical perspective which does not directly cater the core value of service applications where providers' side profitability and growth depend more directly on customer satisfaction[3] in short run and customer loyalty in long run[4]. "Value Driven Design"[5] promotes a movement that is using economic theory to transform the system engineering to better utilize optimization to improve complex design. Enlightened by this ideology, we work towards the foundation of integrating the IT implemenation, business modeling and economic analysis by introducing the *Service Value Broker* (*SVB*) [6] pattern. *SVB* has been already proposed for cloud service brokerage [7] which we foresee as an important characteristic of the optimization of the E-Service composition of [8] E-Service Economics. To the best of our knowledge, there is little work available

A.R. Lomuscio et al. (Eds.): ICSOC 2013 Workshops, LNCS 8377, pp. 163–176, 2014.

in this field. We show that at multiple domains , such patterns may be applied to improve services' performance. Emphasis is placed on situations where mismatch of customer requirements and provided services may occur.These are solved by introducing a *SVB* pattern to improve the resulting composite outputs. In cases where composite services may be improved, the *DSVB* patterns are also introduced which can traverse the entire composition space to achieve improvements. A simulation is performed which shows improvements of multiple metrics based on the broker patterns introduced.

The rest of the paper is organized as follows: Section 2 presents related work of *SVB*. Section 3 models the knowledge foundation of *SVB*. Section 4 demonstrates *SVB* patterns in the Cloud architecture. Section 5 models a two-level E-contract based implementation framework. Section 6 explains the case for the service contract broker. Section 7 provides a simulation to demonstrate the *SVB* pattern in use. This is followed by conclusions with future directions.

2 Related Work

Cloud service brokerage [9] has been identified by many organizations as an key architectural challenge in the Cloud era. In general, most of existing broker research [10,11] focus on using brokers to discover, match, negotiate and select services [12] with best QoS in a service composition. Srikumar et al. [13] adopt brokers to enable grid resource searching and distribution where a broker works mostly as an autonomous agent [14]. D'Mello et al. [15] employ brokers to select qualified services in terms of QoS of SLA for service composition. Loreto et al. [16] use brokers to integrate telephone business and IT world by means of an intermediate layer. Rosenberg and Dustdar [17] use brokers to bridge the difference between heterogeneous business rules. Bichler et al. [18] promote to use brokers to enhance the application level interpretability of electronic commerce.

SVB distinguishes from these approach since it starts from the service contract which covers more issues than SLA. *SVB* is related to services not only on the technological level which covers all three layers of SaaS, PaaS and IaaS of a Cloud architecture, as most SLA based approaches [19], but also on the business level [17,20]. By integrating business services and technology services with value modeling, *SVB* identifies a bigger diagram where it can be successfully applied above QoS assurance [21].

3 Modeling the Foundation of *SVB*

In this section, we model the *SVB* pattern in terms of value, exchange, brokers and composition from the formal view of conceptualization [22].

3.1 Value

Combining the perspective of conceptualization [22] and multiple semantics [23], the concept of value is denoted as: $concept(value)$. In business modeling, an

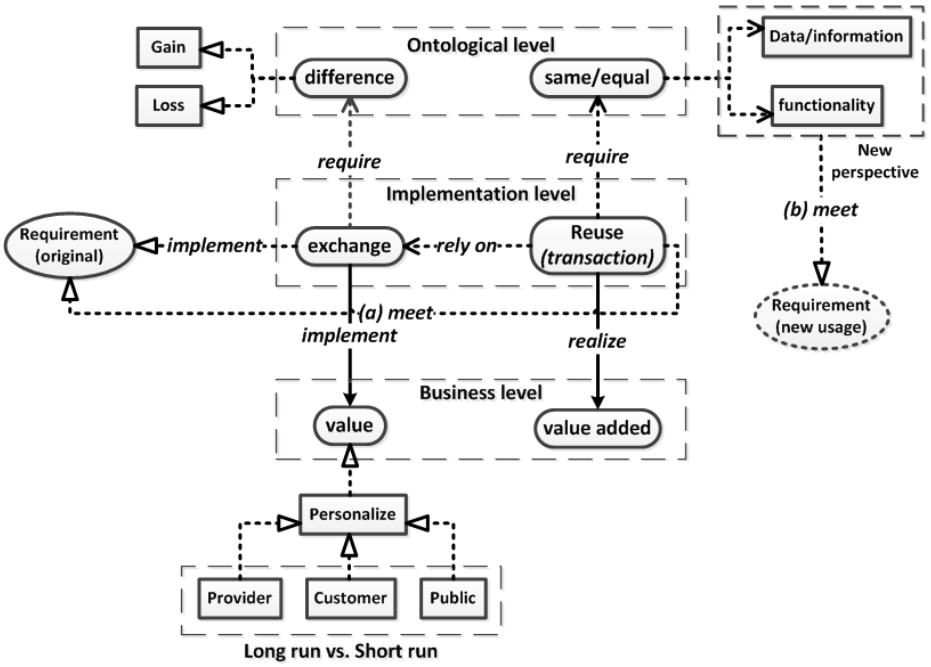


Fig. 1. Value vs. Value added

intuitive expression of value [24] can be found in Fig. 1. Value and value added are modeled along a conceptual route of "Ontological level → Implementation level → Business level". The Ontological level models the essential difference for cognition purpose as "difference vs. same/equal". The implementation level models the basic implementation relationship from "exchange" to "reuse" at transaction level which is distinguished for (a) original requirement and (b) newly identified usages. Newly identified usages demands a difference from existing requirements. The *concept*(*value*) bears several meanings which are denoted as *semantic*{*value*}.

$$\begin{aligned}\text{semantic}\{\text{value}\} = \{&\text{difference (observation}\\ &(\text{object(x)})), \text{existence(object(x))}\}\end{aligned}$$

It means that a meaningful difference can be observed on an existing object in contrast to other objects. The positiveness of the observed difference motivates a business exchange. The observation can either be tangible or intangible.

3.2 Exchange

We improve the explanation from [6,24] as follows: value is realized by exchanging of goods which include information, data, activities, for the purpose of implementation of users' requirements. The existence of the difference of the goods is

required for supporting difference in the value. In a business transaction, users' satisfaction concerning an exchange comes from the positive difference of the expected value of goods. An exchange/transaction should be motivated on both sides of roles of source(A) and target(B). This motivation can exist only when both sides observe a positive value, by subtracting the self evaluated value of the owned object from the self evaluated value of the target one.

3.3 Brokerage

We have introduced the concept of service value broker (*SVB*) in [6]. Here we give the following definitions:

- *Service Value Broker* (*SVB*): driven by a value based goal, when a direct service composition cannot meet some required constraints from the service contract [25] or service level agreement(SLA) such as response time, location, license area, available period, currency format. If the introduction of a intermediate service can help to solve these problems and enable a service composition to be qualified, the introduced intermediate service is a *SVB*.
- *Direct Service Value Broker* (*DSVB*): direct *SVB* is a special type of *SVB* resulting from a composition of services. This composition must bring more value to the stakeholder who introduces the *DSVB*. By value we mean not only monetary value but also non-monetary such as reputation and brand value, etc.

Normally a service transaction is driven by an economical goal marked with a service value scope (SVS) which is denoted as:
SVS = (FBV, OMV)
FBV is the fixed bottom value which is demanded as a profit threshold for a business transaction. OMV is the maximum value which is an open value in most business following the maximization of profit. The SVS is the result of a service composition $composition(x)$ where the involved services are denoted as x. In general, a SVS is denoted as follows:

$$SVS_{ins} = SVS(composition(x))$$

When the restriction $SVS_{ins} \leq FBV$ applies, an independent decision consists of canceling the service. However if other transactions rely on this transaction for constructing an integrated business value, or the transaction is viewed as contributing to accumulated value from a long run economical view [4], then the *SVB* is expected to enable a business transaction. Even those not qualified candidate services could be considered to be transformed into qualified candidates through the introduction of *SVB*.

When the restriction of $SVS_{ins} \geq FBV$ is met, the main goal is the maximization of the value of SVS_{ins}. DSVS can be introduced for this purpose.

Besides introducing new services to replace existing services in a composition, the adjustment of the order of existing services by means of DSVS might also change the value of SVS_{ins}.

3.4 Composition of *SVB* and *DSVB*

In basic situations *SVB* can be introduced to solve direct constraints which are faced in a service composition in the context of a service transaction. *DSVB* is introduced for the optimization of a service system in terms of output value.

In Composed Situations. *DSVB* is the exhaustive search pattern: it requires the traversing of all possible service compositions before a decision is made. *SVB* can function as a transformer which transforms previous not qualified services or service compositions into qualified candidate service compositions which should be considered by *DSVB*. In a real world engineering practice, we can not follow strictly the theoretical conclusion which is drawn here. We can follow a simple process aimed at scaling down the candidate services using matchmaking before a service composition and introduce *SVB* in case that very few candidates are available. *DSVB* will include proactive *SVB* activities only when there is sufficient time left after an initial solution has been found and an optimization is planned.

4 Service Value Broker Patterns: Scenarios and Brokers in Cloud Architecture

When implementing *SVB* or *DSVB*, usually service contracts [25] are required for both locating the mismatching situations and identification of possible solutions. The driving force of applying *SVB* or *DSVB* is to minimally realize expected functionality and optionally attain the highest added value.

We denote the contract on the source end of an exchange as *CS*, the contract on the target end of an exchange as *CT*, the input of *SVB*/*DSVB* contract as *iSVB* and the output of a *SVB*/*DSVB* contract as *oSVB*. There is no requirement that the *iSVB* and *oSVB* belong to the same service since the integration of a parallel set of *SVB*/*DSVB* is allowed. We propose to demonstrate the brokerage within Cloud based on the three-layer architecture of SaaS, PaaS, and IaaS. Fig. 2 demonstrates the brokers in relationship with the three layer Cloud architecture.

4.1 Brokers at the SaaS Layer

There are many kinds of SVB at the SaaS layer. One example is as follows:

Price ($PR \in \mathbb{D}_B$): the price for the service usage is set at "10-20 USD/ month for USA users" while the customers require "5-10 USD/ month for Asia user".
Problem: $PR|_{CS} > PR|_{CT}$
$SVS = (0, \delta(PR|_{CT}, PR|_{CS}))$

Price broker: the price broker is implemented with flexible strategies such as asking a location broker to convey the requests coming from USA to Asia. If the final price after subtracting the cost due to the introduction of the location broker is lower than the original price, the location broker actually implements the role of price broker. There will be other forms of price broker which depend on the specific constraints of the service contracts of both the request and the response sides.
Solution: $(PR|_{CS} = PR|_{iSVB})$ AND $(PR|_{oSVB} = PR|_{CT})$

4.2 Brokers at the PaaS Layer

There are many kinds of SVB at the PaaS layer. One example is as follows:

File/data format ($FF \in \mathbb{D}_B$): requested to provide files with "MS word format" while the provider supplies only files with "pdf or ps format".
Problem: $FF|_{CS} ! = FF|_{CT}$
$SVS = (0, \delta(FF|_{CT}, FF|_{CS}))$
File/data format broker: a service which can convert file format from "pdf or ps format" to "MS word format" has the possibility of playing the broker.
Solution: $(FF|_{CS} = FF|_{iSVB}) AND (FF|_{oSVB} = FF|_{CT})$

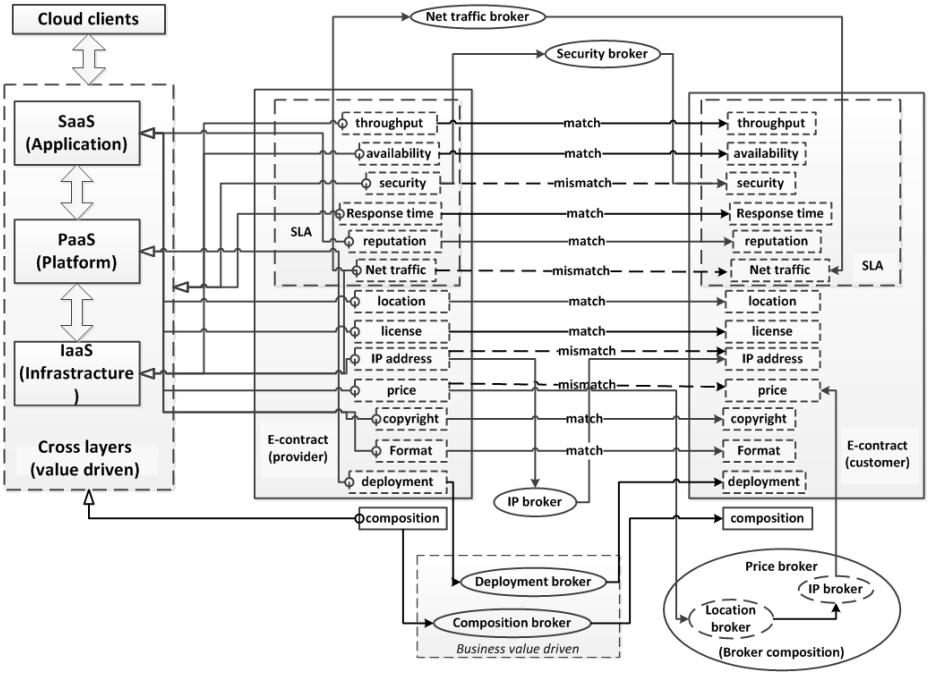


Fig. 2. *SVB* and *DSVB* in the context of Cloud

4.3 Brokers at the IaaS Layer

There are many kinds of SVB at the IaaS layer. One example is as follows:

Network traffic ($NT \in \mathbb{D}_Q$): requests are limited to "band width of 50 MB/ Minute" while customers require "100 MB/Minute".
Problem: $NT|_{CS} < NT|_{CT}$
$SVS = (0, \delta(NT|_{CT}, NT|_{CS}))$
Network traffic broker: a service which can firstly take the request from the customer side of "100MB/Minute", secondarily separate the request into two parallel tasks and finally distribute the two tasks for two services of "band width of 50 MB/Minute ", can play the broker.
Solution: $\sum(RA|_{CS} = RA|_{iSVB})$ AND $(RA|_{oSVB} >= RA|_{CT})$

4.4 The Brokerage Crossing Three Layers

There are also some kinds of SVB which cross different layers of IaaS, PaaS and IaaS. For example, security will crosscut all three layers:

Security limit ($SL \in \mathbb{D}_S$): there will be many security restrictions which might be difficult for a functional service to fulfill.
Problem: $SL|_{CS}! = SL|_{CT}$
$SVS = (0, \delta(SL|_{CT}, SL|_{CS}))$
Security broker: a distributed mode of public-private key architecture can be introduced to enhance the security level of the provided service while not breaking the integrity of the original service. For example, the introduction of audition service and a keying system, can help to avoid a denial-of-service attack (DoS) on the main service.
Solution: $(SL|_{CS} = SL|_{iSVB})$ AND $(SL|_{oSVB} = SL|_{CT})$

4.5 Value Broker

Value broker or *DSVB* is a general form of price broker. It is different from previous brokers which are introduced to solve a mismatch in the conditions for composition, which is demanded rigidly by a service matchmaking process [25]. Value broker is introduced as a mean for the implementation of the optimization process leading to a better business profit for the stakeholder who employs the service based transaction. A glance of value broker enabled maximization of the business solution space is shown in Fig. 3 [24]. In theory, during the implementation of a *DSVB*, all possible service compositions should be considered, including those situations where service compositions are enabled by *SVB* through bridging the functionality mismatching among original services.

Fig. 2 demonstrates a scenario where mismatching situations of security, net traffic and IP between two services represented by E-Contracts are bridged by *SVB*: *Security broker*, *Network traffic broker* and *IP broker*; the optimization for deployment is fulfilled by the *Deployment broker DSVB*. It also shows that brokers can be composed for complex functionalities. For a service transaction comprising more than two parties, there will be the chance to introduce a *Composition broker DSVB* to optimize the organization.

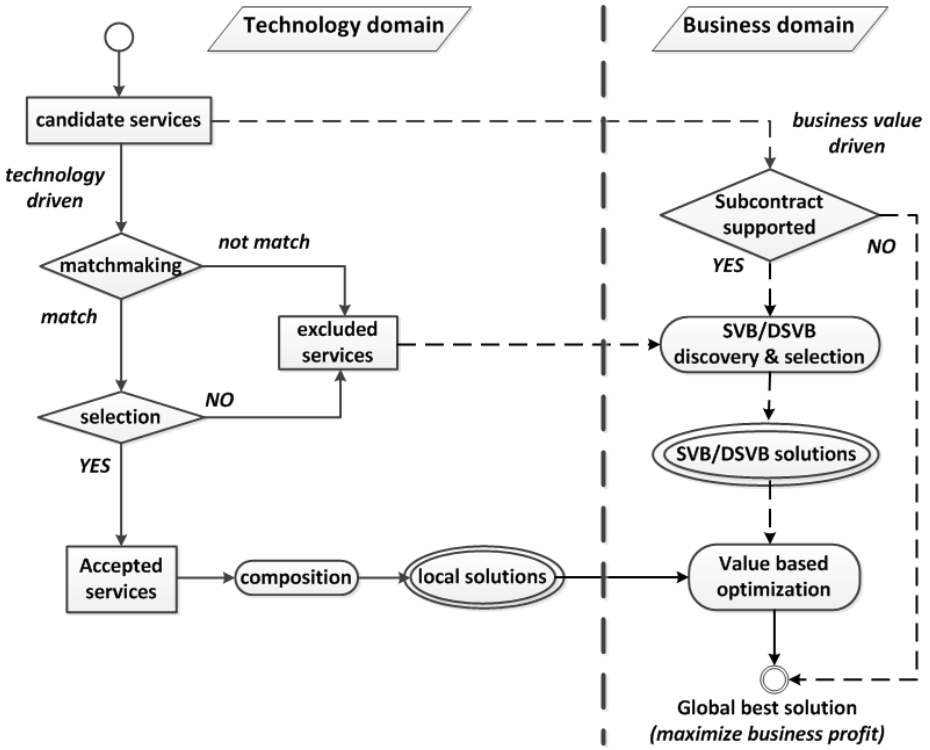


Fig. 3. DVBS enabled maximization of business solution space

5 Two-Level E-contract Based Implementation Framework

In the service ecosystem, due to the interface of the services and their correlation history, the services in the ecosystem will form the composable relation between each other which means that the two services can be used to form a composition to offer added-value for the consumers. As the number of services available for consumers is increasing rapidly, there are many services which offer the similar functionality. For examples,all of "Google Map", "Baidu Map", "Yahoo Map" and "Facebook Map" provide the map related services. These services with the similar functionality will form a specific domain. The service in the same domain can somehow replace each other with some adapters [26]. Furthermore, the providers will publish services into the ecosystem so that the consumers can use the services to fulfill their requirement. Some providers such as Google, Yahoo and Amazon will offer different services in different domains so that they may offer the complete solution for the consumers. Some others will provide a few specific services in the specific domain. Taking Twilio as an example, it focuses on telephony and only offers the Twilio service in the telephony domain for the

consumers. As different providers perform well in different domain, the providers will assign the contract with the others to form a vertical alliance or horizontal alliance to guarantee their core competencies [27]: the providers who provide similar services may assign contracts with each other so that they can get the replace services to increase the fault-tolerance for the consumers; the providers who provide the composable services may assign contracts with each other that they can increase the Qos for the whole composition.

Thus we can get a two-level service contract framework in the service ecosystem which consists of two networks: the service composable network is a directed network in which each node refers to a service and each edge refers to the composability between two services, the direction of the edge refers that the output of the source service can be the input or part of the input for the target service. The provider contract network is an undirected network in which each node refers to a provider and each edge refers to the service contract assigned by two providers.

Fig. 4 demonstrates a two-level service contract network framework for the service ecosystem which consists two networks: the service composable network which refers to the composablity among services, and the provider contract network which refers to the contract relation among providers.

Example: For the illustration shown in Fig. 4 , providers Pa, Pb, Pc, Pd, Pe form the provider contract network based on their contract with each other. Provider Pa offers service S1 and S2, Provider Pb offers services S3, S4 and S5, etc. Service s1, s2, s3, s4, s5, s6, s7, s8 and s9 construct the service composable network and S1, s3, and s6 are similar in the functionality that they form a specific domain.

6 The Case for the Service Contract Broker

6.1 Service Contract Broker for Service Selection

The requirement of the consumer is becoming more complex. Sometime single services cannot fulfill the requirement that they need to select some services to form compositions. If the services are provided by different providers, the providers with a contract can help to guarantee the reputation of the composition. For example, Pd and Pc have a contract while it is not for Pd and Pb, the composition for s6 and s7 will gain a higher reputation than s6 and s5. In this case, the service contract broker will suggest services with higher reputation for the consumers. Even if the services are provided by the same provider, sometimes the QoS cannot meet the consumers' requirement. For example, s1 and s2 can fulfill the consumer's functionality requirement while the price is too high for the consumer. In this case, the service contract broker will help to find the services which are offered by the provider's contractors and then use the service to replace the similar service to fix the mismatch for the consumers. For example, suppose that s3 is much cheaper than s1 and then the broker will use s3 to replace s1 and offer s3 and s2 for the consumers.

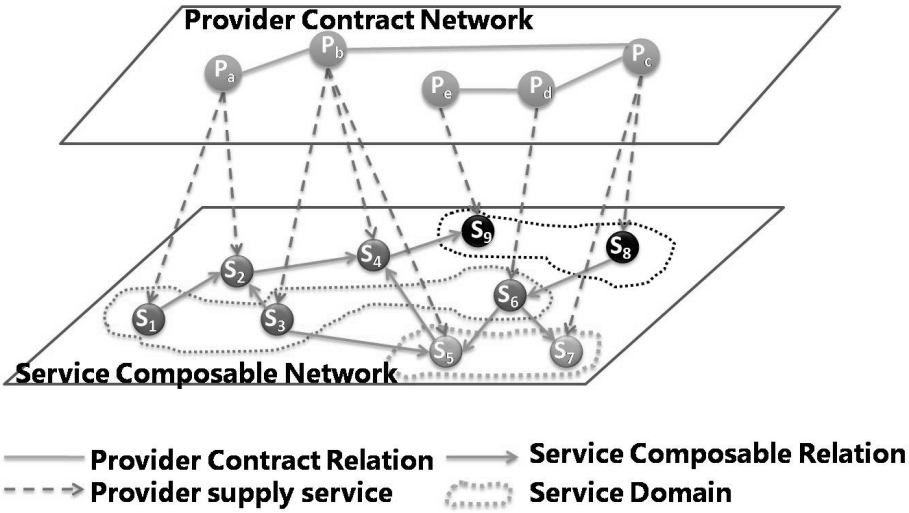


Fig. 4. Two-level service contract network framework

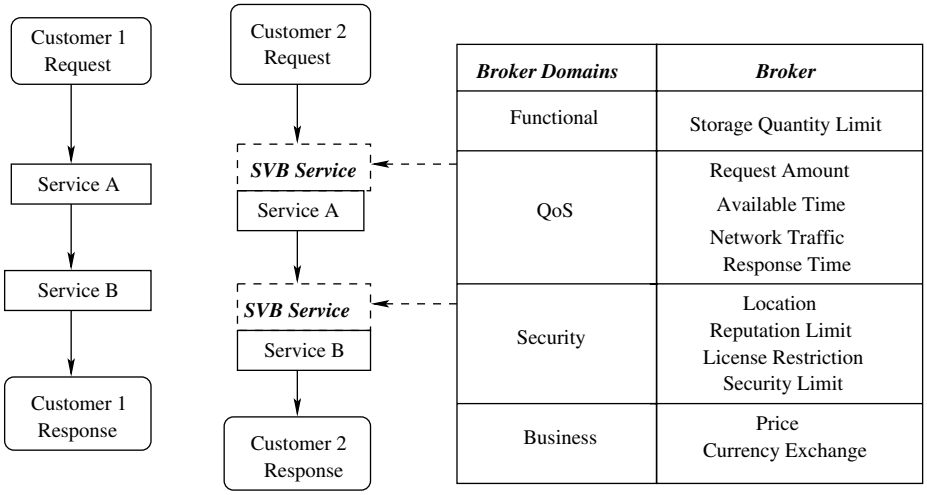
6.2 Service Contract Broker for Service Emerging

For the providers with a strong contract, if the services they offer are not composable, the providers will intend to build an adapter among their services so that their alliance can gain higher competitiveness in the ecosystem. For example, Pb offers service s4 which can be composed with s9 provided by Pe and there is no contract between these two providers. Also Pc offers service s8 which has a similar functionality as s9, however there is huge mismatch between s4 and s8. As Pb and Pc builds a strong contract relation with each other, they may modify the interface of their services to make them composable or create a new service together to bridge their services. In this case, the service contract broker will offer the suggestion for new services. Thus the service contract broker can promote the growth of the service ecosystem.

7 Simulating *SVB*

In order to simulate *SVB* patterns and their effect on customer value [28], we make use of the scenario provided in Fig. 5. While *Customer 1* accesses the sequence of services directly, *Customer 2* makes use of *SVB* brokers to aid in his response. This may come from multiple domains of values.

This scenario is simulated using Monte-Carlo simulations in MATLAB with distributions representing various domains of functional, QoS and business value aspects studied in Section 4. Values such as *response time* and *availability* are modeled as heavy tailed distributions [29]. *Request amount* and *Network Traffic* are modeled with exponential distributions; *Price*, *License Values* and *Security*



Broker Domains	*Broker*
Functional	Storage Quantity Limit
QoS	Request Amount Available Time Network Traffic Response Time
Security	Location Reputation Limit License Restriction Security Limit
Business	Price Currency Exchange

Fig. 5. Scenario comparing two customers

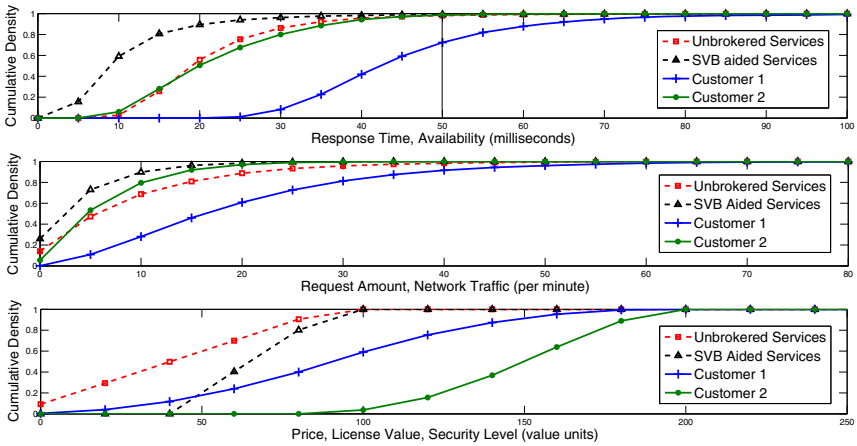


Fig. 6. Monte-Carlo runs of two customers' output behaviors

Levels are drawn from uniform distributions. Note that some brokers such as *Location* and *Reputation Limit* would require a real-world implementation over actual services and are exempted from this analysis. Such a probabilistic model for value is consistent with perspectives of function/QoS/business [29,30].

As observed in Fig. 6, the inclusion of an *SVB* broker improves multiple domains. The response time distribution and network traffic show *lower* values for customer 2. This is traded off with the necessity to pay higher cost values that can provide better security and license values. Though this is a representative example, it can be envisioned as being applicable to real world applications. The

service broker can provide access to valuable upgrades in multiple domains that should be encouraged.

From a *business* perspective, the improved performance due to the introduction of a broker could provide better contractual agreements to a composition of these services. In spite of higher costing services, the tradeoffs can be improved in multiple contractual domains of QoS, security and composition efficiency. Aspects provided by the *DSVB* such as testing and advertisement provide further impetus to the adoption of brokers for business based services.

8 Conclusion and Future Work

This paper presents a value driven design approach that introduces service value brokers as a new form of service design patterns. With this approach, we try to leverage traditional design approaches [2] in order to cope also with business and economical aspects that affect service selection and composition from a value driven design perspective. We present our work towards building an architecture for service brokerage composition in Cloud.

Individually, *SVB*s can be viewed as plain service design patterns [2] when they are used for fulfilling the same functionalities or quality properties. However, from a process perspective, *SVB*s are firstly designed and deployed according to the highest level of economical consideration and profiting context; secondarily, they address specific functionalities and quality requirements. This is usually the reverse process that takes place for plain design patterns. The evaluation of plain design patterns is relatively independent from their combined deployment. However, the evaluation of a *SVB* depends on the integrated business value analysis of the whole project and with relationship to cooperating *SVB*. Moreover, the deployment of *SVB* is expected to be accompanying a value model [5]. In the next steps, we will continue with the labeling of *SVB* with economical properties.

In order to plan and assess a value based composition, we want to explore the constraint space and variability space of the value in *SVB* compositions. We will proceed to the analysis of usability and applicability aspects concerning the adoption of *SVB*s in E-Service contracts.

Acknowledgment. This paper was supported in part by CNSF grant 61162010 and 61363007 and by HNU Research program grant KYQD1242 and HDSF201310. We thank Prof. Zibin Zheng for precious advice.

References

1. Gamma, E., Helm, R., Johnson, R.E., Vlissides, J.M.: Design patterns: Abstraction and reuse of object-oriented design. In: Wang, J. (ed.) ECOOP 1993. LNCS, vol. 707, pp. 406–431. Springer, Heidelberg (1993)
2. Erl, T.: SOA Design Patterns, 1st edn. Prentice Hall PTR, Upper Saddle River (2009)

3. Heskett, J.L., Jones, T.O., Loveman, G.W., Sasser, W.E., Schlesinger, L.: Putting the Service-Profit Chain to Work. Harvard Business Review, 118–129 (July-August 2008)
4. Feldstein, M.: Domestic saving and international capital movements in the long run and the short run. Technical Report 947, National Bureau of Economic Research (1982)
5. Collopy, P., Hollingsworth, P.: Value-driven design. Journal of Aircraft 48(3), 749–759 (2011)
6. Duan, Y.: Modeling service value transfer beyond normalization. In: SNPD, pp. 811–816 (2012)
7. Plummer, D.: Cloud services brokerage: A must-have for most organizations. Gartner, Inc. (2012)
8. Kattepur, A., Benveniste, A., Jard, C.: Optimizing decisions in web services orchestrations. In: Kappel, G., Maamar, Z., Motahari-Nezhad, H.R. (eds.)ICSOC 2011, LNCS, vol. 7084, pp. 77–91. Springer, Heidelberg (2011)
9. Fowley, F., Pahl, C., Zhang, L.: A comparison framework and review of service brokerage solutions for cloud architectures. In: Service-Oriented Computing - ICSOC 2013 Workshops and Ph.D. Symposium (2013)
10. Pan, Z., Baik, J.: Qos broker-based trust model for effective web service selection. In: Proceedings of the 11th IASTED SEA2007, Anaheim, CA, USA, pp. 590–595 (2007)
11. Kumar, P.S.A., Mahadevan, G., Krishn, C.G.: Article: A qos towards dynamic web services recapitulation and selection. International. Journal of Computer Applications 54(4), 12–18 (2012)
12. Shi, C., Lin, D., Ishida, T.: User-centered qos computation for web service selection. In: ICWS, pp. 456–463 (2012)
13. Venugopal, S., Buyya, R., Winton, L.: A grid service broker for scheduling distributed data-oriented applications on global grids. In: Proceedings of the 2nd workshop on Middleware for grid computing, MGC 2004, 75–80 (2004)
14. Qian, Z., Lu, S., Xie, L.: Mobile-agent-based web service composition. In: 4th Intl. conf. on Grid and Cooperative Computing, pp. 35–46
15. D'Mello, D.A., Ananthanarayana, V.S., Thilagam, S.: A qos broker based architecture for dynamic web service selection. In: Proceedings of AMS 2008, pp. 101–106 (2008)
16. Loreto, S., Mecklin, T., Opsenica, M., Rissanen, H.M.: Service broker architecture: location business case and mashups. Comm. Mag. 47(4), 97–103 (2009)
17. Rosenberg, F., Dustdar, S.: Design and implementation of a service-oriented business rules broker. In: CECW, pp. 55–63 (2005)
18. Bichler, M., Segev, A., Beam, C.: An electronic broker for business-to-business electronic commerce on the internet. Int. J. Cooperative Inf. Syst. 7(4), 315–330 (1998)
19. Yu, T., Lin, K.J.: A broker-based framework for qos-aware web service composition. In: EEE, pp. 22–29 (2005)
20. Ferreira, J.E., Braghetto, K.R., Takai, O.K., Pu, C.: Transactional recovery support for robust exception handling in business process services. In: ICWS, pp. 303–310 (2012)
21. Kourtesis, D., Bratanis, K., Friesen, A., Verginadis, Y., Simons, A.J.H., Rossini, A., Schwichtenberg, A., Gouvas, P.: Brokerage for quality assurance and optimization of cloud services: an analysis of key requirements. In: Service-Oriented Computing - ICSOC 2013 Workshops and Ph.D. Symposium (2013)

22. Duan, Y., Cruz, C.: Formalizing semantic of natural language through conceptualization from existence. IJIMT 2(1), 37–42 (2011)
23. Duan, Y.: Semantics Computation: Towards Identifying Answers from Problem Expressions. In: SSNE, pp. 19–24 (2011)
24. Duan, Y.: Value Modeling and Calculation for Everything as a Service (XaaS) based on Reuse. In: Proceedings of SNPD 2012, 162–167 (2012)
25. Duan, Y.: Service Contracts: Current state and Future Directions. In: ICWS, pp. 664–665 (2012)
26. Tan, W., Zhou, M.: Business and Scientific Workflows: A Web Service-Oriented Approach. IEEE Press Series on Systems Science and Engineering. Wiley (2013)
27. Huang, K., Fan, Y., Tan, W., Qian, M.: Bsnet: a network-based framework for service-oriented business ecosystem management. Concurrency and Computation: Practice and Experience 25(13), 1861–1878 (2013)
28. Duan, Y., Kattepur, A., Zagarese, Q., Du, W.: Service value broker patterns: Integrating business modeling and economic analysis with knowledge management (short paper). In: SOCA, pp. 140–145 (2013)
29. Kattepur, A.: Importance sampling of probabilistic contracts in web services. In: Kappel, G., Maamar, Z., Motahari-Nezhad, H.R. (eds.) ICSOC 2011. LNCS, vol. 7084, pp. 557–565. Springer, Heidelberg (2011)
30. Kattepur, A., Benveniste, A., Jard, C.: Negotiation strategies for probabilistic contracts in web services orchestrations. In: ICWS, pp. 106–113 (2012)