T. Saroja Reddy, M.Dharma Reddy, N.Chandra Sekhar Reddy, Sagar Yeruva
International Journal of Engineering Research and Applications (IJERA)
ISSN: 2248-9622 www.ijera.com
Vol. 1, Issue 3, pp.986-989

Effective Integration of External Software Based on Trustworthiness at Selection Time

T. Saroja Reddy [1]
M.Tech (CSE)-Student
CMR College of Engineering & Technology
Hyderabad, AP, India

M.Dharma Reddy [2]
Associate Professor, Dept of CSE
CMR College of Engineering &Technology
Hyderabad, AP, India.

N.Chandra Sekhar Reddy [3]
Professor, Head, Dept of CSE
St.Peter's Engineering College
Hyderabad, AP, India

Sagar Yeruva [4]
Associate Professor, Dept of CSE
St.Peter's Engineering College
Hyderabad, AP, India.

Abstract- In the present competitive environment most of the IT communities tend to go towards off-the –shelf softwares, in which the integration of those external software in project development is challenging , because the execution quality of the software and the trustworthiness of the software provider may be unknown at integration time. Therefore in choosing the SaaS service through reputation systems; however, existing systems rely on ratings provided by consumers. This creates issues like subjectivity and unfairness of the service ratings. This paper explains a framework for automatic selection for software service selection based on quality, cost and trust.

Key Terms-Software-as-a-Service (Saas), Commerercial-off-the–shelf (COTS), Service Oriented Architecture (SOA), service utility, trust, reputation, Service Level Agreement (SLA)

1 INTRODUCTION

The present systems development environment encourages software professional's community to go for commercial off-the-shelf (COTS) software's instead of time-effective and custom "in-house" developed solutions. COTS encourage IT industry like easy to deploy, maintenance-free, and cost-effective. In this scenario through going IT industry tends to move towards Software as a Service (SaaS), where software is delivered on-demand and priced on-use which has been made possible to Internet access which is combined with Service Oriented Architecture (SOA) based solutions which opens the gates to the new domain called "Cloud Computing Environment (CCE)".

But, when we select an outsourced software into our project development it is a challenging and even sometimes risky. We need to check up the performance or quality of the external software which may not be satisfactory at the time of execution. CCE which contains SaaS lowers these kinds of risks. The success of the complete CCE integration depends on the behavior of the provider. Since the software is being delivered as a service, it is hosed at, and maintained by the provider, who fulfills the obligations to the consumer and provides the needed support. He also undertakes the required management and maintenance tasks, and when he generally behaves well, and then the risks of failure remain low. However, the behavior of the service providers is unknown until the service is rendered. The risk of bad behavior cannot be excluded and can have adverse effects on the project outcomes.

We have seen in an empirical study of the risk factors [1] related to the development using external software (Ex: COTS). That paper emphasized that risk reduction at software selection time is negatively correlated with occurrences of most project development-related risks. Therefore we can conclude that selection must be driven by quality constraints, with selection time evaluation of component quality and choice of appropriate service providers which are all essential to successful integration.

Generally the trustworthiness of service provider is commonly measured by their reputation. Reputation systems not only record and track providers' behavior, but can create an incentive for good

behavior by providing consumers with some control over market quality. However existing systems tend to rely on customers' ratings of past service experiences. This creates major issues in terms of subjectivity and rating unfairness.

This paper, we introduce a new framework method for software service selection and rating. The key characteristics of the proposed work are to automate both the selection and the rating of software services which finally increases the objectivity of the service quality reports. This work is grounded on Service Level Agreement (SLA) monitoring to evaluate the derived service model like in [5] and also proposed a novel algorithm which is devised to automate the rating process based on the expectancy-disconfirmation theory form market science [15].

2 MOTIVATION

As explained before, there is no guarantee of service quality at selection time; however, reputation can help in predicting the likelihood of a quality offer to be met. We here have taken three decision making parameters like reputation, quality, and cost. The following figure-1 gives the overview of our proposed model of estimation.

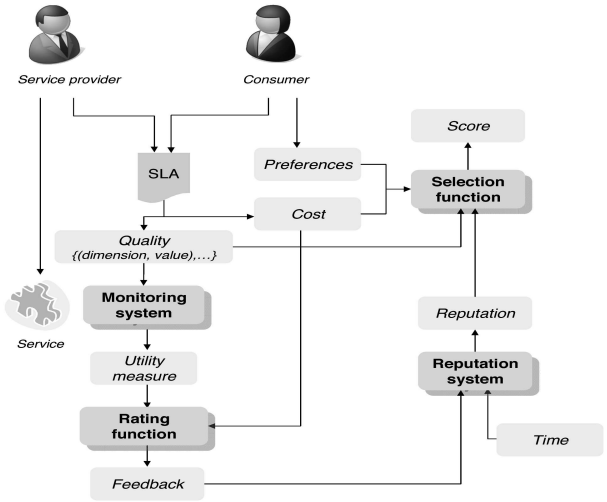

Figure-1: Frame work model of automated estimation

As in the figure-1, ensuring the veracity of reputation reports is also a critical issue. First, feedback can be subjective since it is based on consumers. Consumers may have little incentive to leave positive feedback; they are often more eager to leave negative feedback when they are dissatisfied with the experienced service than to leave positive feedback when they are satisfied. This introduces a bias against positive ratings and leads to unfair reputation reports. We have several contributions on SLA's models like in [7], [8], [9], [10], [11] and [12] which give different parameters for agreement on services. Once SLA has been done we device a selection function that derives a single selection metric out of the reputation of the service as provided by reputation system and the offered quality and cost.

3 AUTOMATION OF RATING AND REPUTATION MODEL

The goal of rating function is to provide objective feedbacks on the selected service without human intervention. We are defining the following model that translates service execution quality into feedback.

3.1 Mathematical Formulation

We use a single scalar metric to quantify quality perception. This metric is basically the utility function of the delivered service. Utility expresses

the conformance of service execution quality to the agreement. The utility function can be considered as the distribution function of the probability that the observed quality meets the agreed quality level during service execution. Thus, the utility function can be estimated from quality monitoring results. We denote by v the utility function of the service. Service quality can be defined as a vector of N dimensions, where N represents the number of quality parameters $QoS_{dim} = Q_1, Q_2 \ldots Q_N$.

The utility function v is defined in [17] as a weighted product of the utilities associated with each parameter Q_i. Compared to a weighted mean, which moderates the impact of low utility levels, a weighted product better reflects the intense impact that a failure in a single quality aspect may have on the overall performance of a QoS [2], [3], [4] sensitive service. For instance, high server response time may cause the client side application to time out before the service is properly rendered. From the consumer's perspective, the impact of a high response time (i.e., low levels in the response-time-related utility) should not be moderated by the permanent availability of the service [13], [14] (i.e., high levels in the uptime-related utility) as it leads to the failure of the service from the consumer's perspective.

v is expressed as follows:

$$\boldsymbol{v} = \prod_{Q_i \in QoS_{dim}} F_{Q_i}{}^{CQ_i} \qquad (1)$$

Where, for each QoS parameter Qi in QoS_{dim}, F_{Qi} is a function that gives the utility associated with the parameter Q_i and the weight $CQ_i \in [0.1]$ reflects how much the user cares about the quality parameter Q_i. CQ_i is user specific; in our model, we consider CQ_i=1 for each parameter Q_i. We also define the function F_{Qi} as the probability for a measured value of Q_i to meet the quality requirement. For instance, the utility of a GenericSaaS is computed as follows:

$$\boldsymbol{v} = F_{uptime} * F_{response\text{-}time} \qquad (2)$$

3.2 Utility Computation

For each quality parameter Q_i, dom(Q_i) denotes the domain of Q_i, (q_i), the agreed (expected) value, and q_i the measured (perceived) value of Q_i. We define the function *Accept* $_i$ as follows:

$$\text{Accept}_i: \begin{cases} dom(Qi) \rightarrow 0.1 \\ qi \rightarrow Accept_i(qi) = \begin{cases} 1 \text{ if } qi \text{ better than or equal to } (qi) \\ 0 \text{ otherwise} \end{cases} \end{cases}$$

The *Accept* $_i$ function follows a Bernoulli distribution.

3.3 Feedback computation

The customer satisfaction $\mathbf{C_{SAT}}$ is defined in [16] as a linear combination of a perception function and a disconfirmation function. It is defined as follows

$$\mathbf{C_{SAT}}(s) = f_p(\boldsymbol{v}) + f_d(\boldsymbol{v}\text{-}\boldsymbol{v}_e) \qquad (4)$$

Where ***v*** denotes the perceived utility, $\boldsymbol{v}_i$ the expected utility, fp the perception function and fd the disconfirmation function. The perception function is described as a concave function considered that the customer is less sensitive to changes in high utility values than to lower ones.

$$\text{FEEDBACK}(s) = f_p(\boldsymbol{v}) + f_d(\boldsymbol{v}\text{-}1) = f(\boldsymbol{v}) \qquad (5)$$

Where *f* is an increasing function defined in [0, 1] and bounded between *f* (0) =0 and *f* (1) =1. *f* should combine the characteristics of both the perception and disconfirmation functions.

4 EXPERIMENTAL STAGE

By making use of the said theoretical approach and combining with the above said mathematical model the simulator takes the input elements like service ID, Cost, Estimated Utility and gives the feedback for the respective service ID as follows

Service ID	Cost	Estimated Utility	Feedback
S1	9	0.88	0.97
S2	95	0.88	0.82
S3	79	0.81	0.75
S4	95	0.70	0.56
S5	9	0.70	0.87
S6	34	0.65	0.80
S7	6	0.42	0.59
S8	91	0.37	0.18
S9	10	0.37	0.52
S10	11	0.17	0.24

Table-1: Experimental inputs and results

We have taken different services like messaging systems (Ex: Yahoo!, GMAIL,way2sms,……..etc), Internet providing systems like(Ex: Sify,BSNL, AirTel,…….etc), different programming language systems (Ex:C, C++, JAVA, .Net,……etc) and finally specific application systems required for the system to run are considered and are given some unique identification number for each service providers. For the respective service based on the reputation we have estimated Cost and Estimated Utility through SLA.

5 SELECTION PROCESS

As we discussed before, services are compared against three criteria (quality, cost, and reputation) before any selection is made. In the previous section of experimental results we have seen automatic feedback given by the simulator for the respective inputs for some service IDs. Now we need to select the service. We investigate the ways in which quality, cost, and reputation can be combined in support of decision making. We use a simple method that aggregates the three parameters in to a single ranking metric in three steps like

1. **Match making:** It compares service offers against user requirements. All of the offers which do not meet user requirements, commonly expressed in terms of quality and cost constraints, are ignored.

2. **Evaluation:** All eligible services offer a quality level that is equal to or higher than requested and come at affordable costs. We will thus evaluate service offers in terms of the gain in quality and cost is proposed.
3. **Ranking:** This results to **SCORE(s),** the ultimate selection metric. The service with the highest **SCORE(s)** is then selected.

6 CONCLUSION

In this paper, we have discussed the major problems in the present IT industry and the main advantages of COTS for the present systems which mainly help to deploy, maintenance-free, and cost-effective in SOAs. In this perspective we have presented an automatic quality and reputation based framework for service rating and selection. Even though some models have come but none of them have considered the automation of the service rating process.

This framework method which allows us to rate a feedback to be assigned to a delivered service that objectively reflects the satisfaction and dissatisfaction and quality. The mathematical model and method has been designed to assist customers in selecting the most appropriate service offering considering quality and cost constraints. Reputation is used to predict the credibility of the quality offer and the conformance of this offer to the delivered quality. We used a service ranking method that aggregates the quality, cost, and reputation parameters into a single metric that is used to evaluate service offerings against each other. This work aims to design the system that supports the integration of software service in application development and provisioning projects.

7 REFERENCES

[1] J. Li, R. Conradi, O.P. Slyngstad, M. Torchiano, M. Morisio, and C.Bunse, "A State-of-the-Practice Survey of Risk Management in Development with Off-the-Shelf Software Components," IEEE Trans. Software Eng., vol. 34, no. 2, pp. 271-286, Mar./Apr. 2008.

[2] J. Anselmi, D. Ardagna, and P. Cremonesi, "A QoS-Based Selection Approach of Autonomic Grid Services," Proc. Workshop Service-Oriented Computing Performance: Aspects, Issues, and Approaches, 2007.

[3] L. Zeng, B. Benatallah, A.H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-Aware middleware for Web Services Composition,"

[4] IEEE Trans. Software Eng., vol. 30, no. 5, pp. 311-327, May 2004.

[5] L.-H. Vu, M. Hauswirth, and K. Aberer, "QoS-Based Service Selection and Ranking with Trust and Reputation Management," Proc. 13th Conf. Cooperative Information Systems, 2005.

[6] J. Skene, F. Raimondi, and W. Emmerich, "Service-Level Agreements for Electronic Services," IEEE Trans. Software Eng., vol. 36, no. 2, pp. 288-304, Mar./Apr. 2010, http://doi.ieeecomputer

society.org/10.1109/TSE.2009.55.

[7] J.R. Douceur, "The Sybil Attack," Proc. First Int'l Workshop Peer-to- Peer Systems, 2002.

[8] J. Skene, A. Skene, J. Crampton, and W. Emmerich, "The Monitorability of Service-Level agreements for Application- Service

[9] Provision," Proc. Sixth Int'l Workshop Software and Performanc, pp. 3-14, 2007.

[10] BelGOnet, "Service Level Agreement," http://www.belgonet.com/website/UK/Service_ Level_Agreement_UK.pdf, 2008.

[11] EZSM, "EZSM Service Level Agreement," http://www.easyservermanagement.com/sla.php , 2008.

[12] Amazon, "Amazon s3 Service Level Agreement," http://aws.amazon.com/s3-sla/, 2007.

[13] Intacct, "Intacct Buy with Confidence," http://us.intacct.com/downloads/08datasheets/DS_Buy_with_Confidence.pdf, 2008.

[14] H. Ludwig, R.P. King, and A. Keller, "Web Service Level Agreements," http://www.research.ibm.com/wsla/sampleoutsourced.wsla, 2002.

[15] Y. Wang and J. Vassileva, "A Review on Trust and Reputation for Web Service Selection," Proc. 27th Int'l Conf. Distributed Computing Systems Workshops, 2007.

[16] T. SaaS, "Trust Saas: Putting the Trust in Software as a Service (SaaS)," http://trustsaas.com/, 2008.

[17] R.L. Oliver, "A Cognitive Model of the Antecedents and Consequences of Satisfaction Decisions," J. Marketing Research, vol. 17, pp. 460-469, Nov. 1980.

[18] J. Xiao and R. Boutaba, "Assessing Network Service Profitability: Modeling from Market Science Perspective," IEEE/ACM Trans.Networking, vol. 15, no. 6, pp. 1307-1320, Dec. 2007.

[19] V. Poladian, J.P. Sousa, D. Garlan, and M. Shaw, "Dynamic Configuration of Resource-

[20] Aware Services," Proc. 26th Int'l Conf. Software Eng., 2004.