Trustworthy Service-Oriented Computing

Chung-Wei Hang
chang@ncsu.edu
Dept. of Computer Science
North Carolina State University
Raleigh, NC 27695-8206, USA

1. THE PROBLEM

In service-oriented computing environments, computing resources are managed as *services*, which can be used directly or composed into larger services. Service-oriented architecture has been widely adopted in modern distributed environments such as for cloud computing. However, the problem of finding desired services has arisen. Finding desired services can be divided into two sub-problems: *service discovery* and *service selection*. The former one emphasizes how to find services that match consumers' requirements. The latter focuses on how to select *best* matched services. Service discovery usually finds services based on static functional attributes (e.g., service descriptions), whereas service selection tends to capture the dynamism of nonfunctional properties. For example, suppose a traveler is looking for flight tickets from Raleigh to Budapest. Service discovery returns itineraries provided by various airline companies. Service selection, on the other hand, selects best quality itinerary in terms of in-flight service, delay record, etc.

Traditional service discovery approaches, such as, *Web Service Definition Language* or *WSDL*, and *Service level agreement* or *SLA*, describe the functional configurations of services. However, these approaches lack mechanisms to monitor and track the nonfunctional properties like quality of service (QoS) dynamically. For instance, suppose an United flight delays due to technical difficulties, how this affects the airline company the consumers choose next time? The QoS assessments should be able to reflect the expected outcome of future behavior and affect consumers' willingness to select that service, even though the service matches their requirements.

2. QOS-BASED SERVICE SELECTION

We introduce the idea of QoS-based service selection approach to address the problem of selecting services based on both functional and nonfunctional properties. There are two main problems to be solved.

The advantage of service-oriented computing is that we can compose services to create new ones. This is called *service composition*. Much attention on service composition focuses on lower-level solutions, such as, BPEL, OWL-S, π-calculus, and Petri nets [5]. These methods capture the composition configurations, but, similar to WSDL, fail to take nonfunctional properties into consideration. Services are composed into larger services. However, these underlying services may not be directly exposed to the consumers. Service composition can be divided into many scenarios [4] and these scenarios can be nested. These scenarios make QoS metrics hard to collect and evaluate. For example, a traveler books an itinerary from a travel agent without knowing which hotel agent is behind. Thus, service selection becomes more complicated because the consumers may not even know with whom they are interacting. Existing service selection approaches deal with service composition poorly because they mostly either not consider service composition, or assume the composition information is fully observable. Therefore, how to collect QoS metrics, and how to evaluate the underlying services behind composition must be addressed in our service selection solution.

Another challenge is, even if a service can be evaluated based on past experience, consumers may lack past experience of unknown parties. One common solution is to bootstrap the unknown parties by assigning initial assessments as new comers. Better solution is to introduce referral networks. One may ask others for referrals of an unknown party, which is called the target. The referrals contain either direct information with the target, or further references if the referrers have no experience themselves. The initial party can follow the *referral chains* until certain criteria are met, say, until a certain depth. After collecting the referrals, the initial party can *aggregate* all information gathered as the experience of the target, also evaluate the *sociability* of referrers. The sociability is the ability of providing accurate referrals.

3. SOCIAL TRUST MODEL

Trust modeling in artificial intelligence provides us a promising solution to above questions. Trust is a basis of interactions, indicating the relationships between parties in large, open systems. Two parties must trust each other sufficiently to be willing to carry out desired interactions. In a service-oriented context, a party Alice trusts another party Bob, because Alice expects Bob will provide desired service. In general, an ideal trust model should contain following functionalities: *trust representation*, *trust propagation*, and *trust update*.

The trustworthiness of a party should be represented as not only a *probability*, but also the *confidence* of the probability. An ideal trust representation should satisfy: (a) the confidence goes up as the evidence increases given a fixed probability, and (b) the confidence drops if conflicts occur given a fixed amount of evidence.

Trust propagation defines how trust information is propagated. There are two basic cases. First, how indirect trust information should be discounted? For example, Alice trusts Bob who trusts Charlie. Alice should not consider the trust information of Charlie from Bob totally. Instead, Alice should discount Bob's trust in Charlie by her trust in Bob. Second, how trust information should be combined? For example, Alice collects trust information of Dave from both Bob and Charlie. A trust model should define how trust information from different sources is aggregated.

The third component is trust update. As we gain more experience with the target, trust should be updated in the way that updated trust can predict the target's behavior more accurately. For example, Alice asks Bob for referrals of Charlie. When Alice has better knowledge of Charlie, how she updates her trust in Bob about his sociability? Generally, given estimated trust and actual knowledge, trust update defines how accurate the estimation is.

Our previous work [1, 2] provides a complete solution to trust modeling. We adopt the trust representation from [6, 7], which defines trust in both evidence and belief space. For example, the trust in evidence space $\langle r, s \rangle$ represents how much good and bad evidence we have with the target. The probability is defined by $\frac{r}{r+s}$. In belief space, $\langle b, d, u \rangle$ corresponds to *belief* (belief of trust), *disbelief* (belief of distrust), and *uncertainty*, respectively. The trust can be translated between evidence and belief spaces. The definition of uncertainty satisfies the two requirements of confidence. We also define trust update by comparing the difference of probability-densities of the estimation and the actual trust. Finally, our trust model provides three trust propagation operators: *concatenation*, *aggregation*, and *selection*. The concatenation operator defines how indirect information should be discounted, whereas the aggregation operator is used to combine trust evidence from different sources. The selection operator exempts trust propagation from *double counting*. Additionally, our trust model is verified via simulations and social network data.

4. QOS-BASED TRUSTWORTHY SERVICE SELECTION

We aim to provide a QoS-based trustworthy service selection method based on our trust model. There are three main components as follows:

1. Developing an ontology that include classes, relationships, and attributes required to characterize services and their uses in service-oriented environments.

2. Formalizing rich service composition models built on trust attributes specified in the above ontology.

3. Developing approaches for agents to monitor and explore desired service compositions dynamically.

We refine and enhance an existing QoS ontology from [3] to fit it into our approach. This ontology will be able to capture SLAs as well as the requirements of consumers and advertisements from providers. Both domain-independent and domain-specific QoS properties can be defined in our ontology. We model the service-oriented environments by a directed graph. The graph can capture the relationships between services in service composition. Then, QoS properties are monitored and collected from direct experience and indirect evidence (i.e., referrals). The QoS assessments are represented as trust. The trustworthiness of a QoS attribute can be inferred by trust propagation. Also, we can further evaluate the QoS properties, by comparing the QoS metrics and SLAs, and the sociability of referrers by trust update. Knowing the sociability can yield more accurate trust information from referrals. Finally, we will apply *multiattribute utility theory* for decision-making, based on the trustworthy QoS assessments.

5. CHALLENGES

Our main challenge is how to capture the relationships in service compositions so that the trustworthy QoS assessment can accurately reflect the QoS of services. For example, a traveler books an itinerary from a travel service, which interacts with a flight service, a hotel service, and a car rental service. Suppose the availability of the car rental service is not satisfiable. This ends up with bad availability of the travel service. Given the fact that the traveler is not aware of the services behind, an appropriate mechanism is needed in order to punish the car rental service, and the travel service (because it selects the car rental service), rather than the flight and hotel services.

6. CONCLUSION

This work aims to provide a QoS-based trustworthy service selection model in service-oriented environments. The model provides an ontology to capture consumers' requirements and providers' advertisements dynamically. We formalize a graphical service composition model to capture the relationships between services, develop approaches for consumers to monitor and explore desired services and service compositions. Our trust model, built on [1, 2], estimates trustworthiness of services in term of QoS properties, from both direct experience and indirect referrals for consumers to select desired services.

7. REFERENCES

[1] C.-W. Hang, Y. Wang, and M. P. Singh. An adaptive probabilistic trust model and its evaluation. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1485–1488, 2008.

[2] C.-W. Hang, Y. Wang, and M. P. Singh. Operators for propagating trust and their evaluation in social networks. In *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (To appear)*, 2009.

[3] E. M. Maximilien and M. P. Singh. A framework and ontology for dynamic web services selection. *IEEE Internet Computing*, 8(5):84–93, Sept. 2004.

[4] D. A. Menascé. Composing web services: A QoS view. *IEEE Internet Computing*, 8(6):88–90, 2004.

[5] N. Milanovic and M. Malek. Current solutions for web service composition. *IEEE Internet Computing*, 8(6):51–59, 2004.

[6] Y. Wang and M. P. Singh. Trust representation and aggregation in a distributed agent system. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, pages 1425–1430, Menlo Park, 2006. AAAI Press.

[7] Y. Wang and M. P. Singh. Formal trust model for multiagent systems. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1551–1556, Detroit, 2007. IJCAI.