

Trustworthy Service Discovery for Dynamic Web Service Composition

Yukyong Kim¹, Jong-Seok Choi¹ and Yongtae Shin¹

¹Dept. of Computer Science and Engineering, Soongsil University
Seoul, 156743 –Korea

[e-mail: yukyong@ssu.ac.kr, jschoi@ssu.ac.kr, shin@ssu.ac.kr]

*Corresponding author: Yongtae Shin

*Received September 25, 2014; revised December 19, 2014; revised February 4, 2015;
accepted February 27, 2015; published March 31, 2015*

Abstract

As the number of services available on the Web increases, it is vital to be able to identify which services can be trusted. Since there can be an extremely large number of potential services that offer similar functionality, it is challenging to select the right ones. Service requestors have to decide which services closely satisfy their needs, and they must worry about the reliability of the service provider. Although an individual service can be trusted, a composed service is not guaranteed to be trustworthy. In this paper, we present a trust model that supports service discovery and composition based on trustworthiness. We define a method to evaluate trust in order to discover trustworthy services. We also provide a method to perform trust estimation for dynamic service composition, and we present results of two experiments. The proposed model allows for service requestors to obtain the most trustworthy services possible. Our mechanism uses direct and indirect user experience to discover the trustworthiness of the services and service providers. Moreover, composing services based on quantitative trust measurements will allow for consumers to acquire a highly reliable service that meet their quality and functional requirements.

Keywords: Trustworthy service discovery, dynamic service composition, QoS evaluation, trust management

This work was supported by the ICT R&D program of MSIP/IITP, Korea. [10044556, The development of network technology in defense infrastructure for high quality convergence service]. This work was also supported by the BK21Plus Software Security Program, Soongsil University funded by the Ministry of Education, Korea and National Research Foundation of Korea.

1. Introduction

A service-oriented architecture (SOA) is a very popular architecture paradigm that can be used to design and develop modern distributed systems such as cloud computing. SOA is also increasingly used in enterprise information systems, especially in the form of Web services, because it can help realize business goals that require easy, flexible integration with legacy systems, streamlined business processes, reduced costs, innovative service to customers, and agile adaptation and reaction to opportunities and competitive threats [1]. In order to realize the potential of service oriented computing (SOC), an SOA should be designed to overcome the many possible challenges encountered in the enterprise, including the need for complex distributed services, managing business processes, ensuring transaction Quality of Service (QoS), complying with agreements, and leveraging different computing devices such as personal computers and cell phones [2].

In order to build a service-oriented application, an application developer, or service requestor, can select services from a variety of providers on the Web. Since there is an extremely large number of potential services with similar functionality, service requestors need to differentiate between them. The only differentiating factor between similar services may be their non-functional properties, which can be considered to be part of the criteria for service selection. In this context, it is a challenge to properly select the right services. Service requestors have to decide which services satisfy their needs, and they have to worry about the trustworthiness of the service provider. Trust is a non-functional property that can be used as a criterion for service selection. A service requestor, or trustor, may select a service from a service provider, the trustee, based on trustworthiness. Thus, trust can help requestors make their service selection decision.

Trust is a vast, subjective field that has been defined in the context of e-commerce, Web services, and peer to peer (P2P) networks. For SOA and Web services, Chang et al. have defined trust as “the belief the trusting agent has in the trusted agent’s willingness and capability to deliver a mutually agreed service in a given context and in a given time slot” [3]. Trust management is of paramount importance to reduce the risk involved in transactions between services and service consumers. The challenge in implementing trust mechanisms is to find good models from which trust can be derived directly from use experience, recommendations from third parties, and social relationships. In particular, such models aim to develop mechanisms that can provide robustness against attacks such as misleading recommendations, reentry and Sybil attacks. Therefore, an effective trust model is important to address such issues.

However, it is difficult to solve this problem using the current service discovery techniques such as Universal Description, Discovery and Integration (UDDI). Information for Web services in the UDDI registry cannot ensure that these would be reliable and usable. According to the Web services report in [4], 48% of the production UDDI registry (tModels tested only) has unstable links. Recent research on trust modeling provides us with a promising starting point to find a solution for service selection [5]. Trust is the key basis of interaction in an open setting, indicating that there is a relationship between the parties involved. For example, in SOC environments, a party A may trust another party B, because A expects B will provide a service of a certain desired functionality and quality. We define trust-aware service selection as selecting the desired services based on the trust placed in their ability to deliver specified values of specified qualities.

Intuitively, the trustworthiness of a service should be estimated according to both direct and indirect experience. Direct experience indicates the service had been previously received with a certain quality, whereas indirect experience indicates that derived from referrals by peers. It is not straightforward to estimating trust from direct experience with a service because some services may not directly expose details of their composition to their consumers. Although individual services can be trusted, composed services are not guaranteed to be trustworthy. A consumer may interact with a composite service without knowing much about the qualities of the services that underlie it, and in such a case, evaluating the trustworthiness of the service is nontrivial. In order to select a service, service compositions should be modeled with respect to how the quality of a component service can affect the whole composition. In this case, we adopt the concept of a social relationship network for e-commerce. According to the e-commerce social relationship network, the model from the buyer's point of view calculates the trust value list of every seller. For Web services, every consumer has many services available, and every consumer is willing to refer to the reputation reported by other consumers, which is the theoretical basis of our approach.

In this paper, we first construct a model from the direct experience of a service consumer. The model represents attributes of quality that are observable from the consumer's point of view. Then we present an approach that models service compositions via graph theory in terms of partially observable factors. The approach captures the dependency for providing a good quality of service between elementary and composite services. It shows how a composite service's quality depends upon the quality of its elements. Those two models are combined to provide a quality distribution for the services, including information on how much each elementary service contributes to the composition. The rest of the paper is organized as follows: Section 2 briefly describes some previous work with respect to the trust computing model, and Section 3 defines the concept of trust and formalizes the problem from point of view of the service consumer. Section 4 specifically introduces a method to evaluate trust derived from each service, and it also describes our approach to estimate the trust value for a composite service. Section 5 gives experimental results. Finally, Section 6 summarizes and proposes the work for the next phase.

2. Related Work

Fig. 1 presents different trust-based service selection approaches [6]. In terms of the direct experience approach, requestors build a trust profile for services after utilizing them. However, since there is a need to trust services before they are executed, a Trusted Third Party (TTP) approach enables consumers to consult a trusted third party to determine the trustworthiness of the services. For the indirect experience approach, reputation is a public opinion of the characteristics of an entity, and it represents the collective evaluation of that entity. Recommendation systems aggregate recommendations and match recommenders with others searching for recommendations. Finally, a referral approach is a decentralized approach based on software agents and communities, where agents can assist each other in finding services by giving referrals to services that were useful for them. Therefore, each agent builds the trust of other agents according to the perceived quality of the services and the referrals that guide them to the services. In a matchmaking TTP approach, a service description is matched with a requestor's request and with the trust preferences. The hybrid approach involves a combination of different approaches that aim to improve the weaknesses of some of the approaches by combining them with other approaches. The automated trust negotiation approach builds mutual trust between service requestors and service providers. In this approach, trust is

assessed in two directions: a requestor trusts a service and the service trusts the requestor. The trust negotiation approach depends on the disclosure of digital credentials between the two parties.

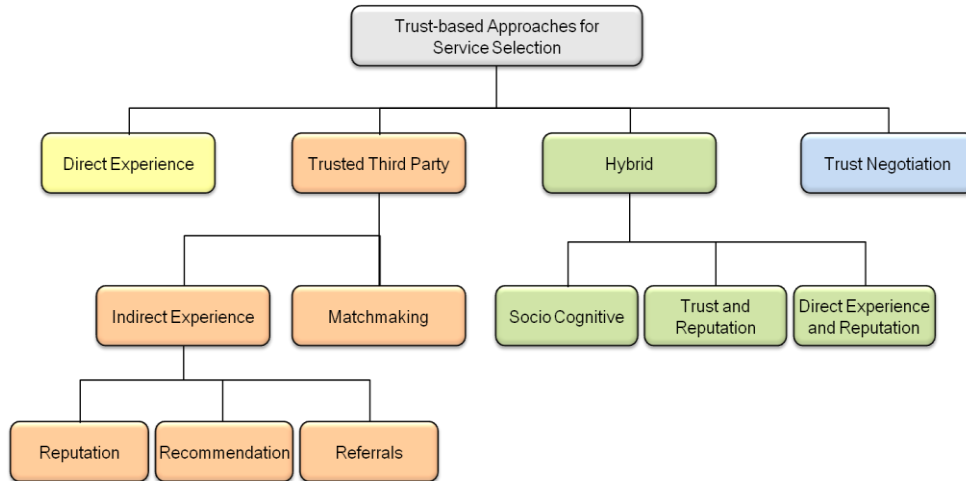


Fig. 1. Approaches for Trust-Based Service Selection

Researchers investigating trust have also attracted great attention to SOC. However, the literature on SOA trust is still immature. There are some the Web service security standards that ensure hard security mechanisms for SOA applications that have been addressed by IBM and Microsoft. These organizations cooperate to define a unified approach for managing message security exchange in a Web Services environment [2]. Although Web services security technologies, such as WS-Security and WS-Trust, increase security in Web Services, they cannot determine the entities that have hidden motives or other trust issues, such as detecting unfair ratings, determining the level of trustworthiness of entities, rewarding positive behavior, punishing malicious behavior, and reputation bias [7].

In [8], Wu et al. model a consumer's assessment of the quality of a service using a naive Bayesian network. They apply a fuzzy representation to express the levels of the capabilities of the service. Their approach enables consumers to estimate the overall quality. Lin et al. select services according to a consensus of the group preferences in order of various qualities [9]. They use fuzzy logic to resolve the conflicts between the subjective interpretations of the service qualities from each consumer. In [10], Yue et al. apply Bayesian networks to model the relationships between elementary services, and their approach constructs web service Bayesian networks (WSBN) based on the invocations between the services. In [11], Paradesi et al. build a trust framework for web service compositions. They adopt the trust representation from [12] and introduce operators to combine trust in different types of service compositions, including sequence, concurrent, conditional, and loops. Maximilien and Singh develop a trust-aware approach that selects services based on a well-defined ontology which provides a basis for describing consumers' requirements and providers' advertisements [13]. The ontology enables consumers to define nonfunctional properties, but this approach does not take into account service composition. Wang and Singh developed a trust model for multi-agent systems that formalizes how agents map evidence to trust and vice versa [12]. In [14], Artz and Gil compare several definitions of trust in different research areas in the field of computer science. In particular, the authors discuss the relevance of trust and the semantic Web and point out some unique trust management challenges in that area. Fernandez-Gago et

al. perform a trust management survey focusing on wireless sensor networks [15], and their survey is an overview of existing trust management solutions for ad-hoc and the peer-to-peer (P2P) wireless sensor networks. A few surveys focus on reputation-based trust management systems. For instance, in [16], Marti and Garcia-Molina exploit a taxonomy technique to classify different reputation-based trust management systems. In [17], Jøsang et al. discuss general ideas of trust (e.g., trust classes and trust purpose), explain the overlapping notions between trust and reputation terms, and compare a few trust models. In [18], Silaghi et al. investigate whether existing trust management approaches can be applied to Grid environments, and a few guidelines are given in the survey that may be useful to future research and the development of trust management systems in Grids. In [19], Wang and Vassilev present a systematic review of several trust and reputation systems in which they classify these systems into three categories including centralized versus decentralized, persons/agents versus resources, and global versus personalized.

A novel service composition algorithm is proposed in [20]. The algorithm models the service composition as a multi-domain scheduling problem with minimal service resources and time constraint. They define each service as an exclusive resource during its execution period. By computing the inter-domain communications and available services in each domain, the domain with optimal utilization rate is obtained to arrange services. In [21], a framework for a Trust-based Dynamic Web service Composition is presented. The framework not only uses functional and non-functional attributes provided by the Web service description document but also filters and ranks solutions based on their trust rating. They define a method to calculate a trust rating per service using Centrality measure of Social Networks. In [22], Wang et al. formulate the problem of service composition and service binding as a multi-objective optimization (MOO) problem, minimizing the service cost, while maximizing the quality of service (QoS) and quality of information (QoI). They develop a trust-based service composition and binding protocol, and demonstrate that the trust-based scheme outperforms the counterpart non-trust-based scheme. Wu et al. propose a trust-based service composition and optimization method in [23]. They define the trust of service composition in terms of the trust of component service selection processes, composition processes and optimal binding plans. They use filtration, interface-based service clustering, and trust evaluation to guarantee the trustworthiness of service selection and composition.

Trust management is one of the most important issues in the area of information security, and several surveys have been conducted in this regard. Our work presents how to evaluate the trustworthiness of services that are elementary components of a composite service. We provide a systematic way to represent qualities of service compositions via operators that correspond to typical ways of composing services. Our contribution is to provide a method that can handle service selection according to trustworthiness in the context of service compositions.

3. Problem Formulation

3.1 The Concept of Trust

Trust is widely accepted as a major component of human social relationships. In general, trust is a measure of confidence that an entity will behave in an expected manner, despite the lack of ability to monitor or control the environment in which it operates. OASIS (Organization for the Advancement of Structured Information Standards) defines trust as “the characteristic that one entity is willing to rely upon a second entity to execute a set of actions and/or to make assertions about a set of subjects and/or scopes” [24].

However, it is difficult to objectively judge whether or not a particular entity is trustworthy. Therefore, the concept of trust is incorporated into a trust management system that establishes monitors and adjusts trust relationships among the participants. The core concept of trust management is represented by a trust model that defines primary factors of trust relationships and describes how to calculate the resulting trust values.

To establish a trust model for SOA, we consider trust as the belief that one agent has in another agent's capability to deliver a quality of service in a given context and in a given time slot. We assume that an agent can be an abstract entity in a distributed network and can represent an ordinary service user, a service, or a service provider. Trust is formalized by the concept of a trust relationship between agent a_i and a_j from the set of agents $A = \{a_1, a_2, \dots, a_i, \dots, a_j, \dots, a_n\}$, and it is expressed as a binary relation $R \subseteq A \times A$. The trust relation $a_i R a_j$ represents a directed link between agent a_i and a_j in a directed graph which is called a trust graph (TG) denoted by an ordered pair $TG = (A, R)$.

We formulate the knowledge as trustworthiness in a specific time and context. That is, the knowledge is quantified according to past experience during a specific time and context. To establish a trust model, we use the time and context dimensions. The time dimension T is a set of time values t_i when the interaction between the agents took place, and the context dimension C contains a set of propositions that can be formulated as a sequence of possible events.

A trust value is formulated when an element $r \in R$ has a degree of trust from a domain set of possible trust values D . When $r_{i,j} = a_i R a_j$ for a_i and a_j in A , the trust value for $r_{i,j}$ is defined as the trust degree of an agent a_i toward agent a_j in time t and in context c , and it is represented by $\tau_{i,j}(t, c) \in D$ where $t \in T$ and $c \in C$. We will then describe $\tau_{i,j}(t, c)$ using direct and indirect measurements in the next section. **Fig. 2** shows an example of a trust graph in some time and context. On the trust graph, the directed links between agents have weight values $\tau_{i,j}$ from the domain set D .

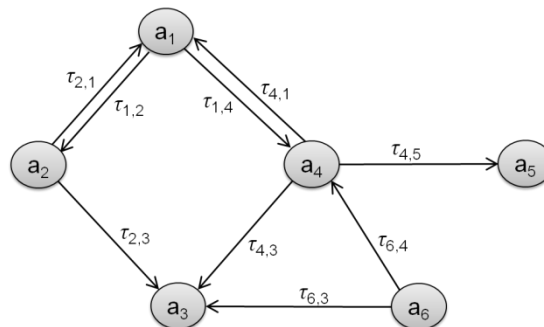


Fig. 2. Trust graph during some time and in some context

Trust has several main characteristics that remain true regardless how trust is defined, as discussed below:

- Trust is reflexive. Since an agent trusts himself/herself implicitly, trust relationships are reflexive.
- Trust is not symmetric. Trust level is not identical between two parties. For example, A may trust B 100%, however, B may not necessarily feel the same way about A; B may only trust A 50%. Since an agent can trust someone who does not trust the agent, it's a one-way relationship that may or may not be mutual.
- Trust is transitive. Although trust is not perfectly transitive in the mathematical sense, there is, however, a notion that trust can be passed between people. In the

mathematical sense, the first agent might trust a second one who trusts a third one, but trust between the first and third agents may or may not propagate transitively. On the other hand, when encountering an unknown person, it is common for people to ask trusted friends for opinions about how much to trust this new person [5, 25]. Thus trust can be passed along a chain of trusting people. This logic also supports the use of a single value to represent trust in a person and trust in their recommendations about other people. From this point of view, trust is propagative, and we introduce indirect experience from referrals by agents.

- Trust is dynamic. Trust is very changeable. Like a person's fickle mind, it may decrease, increase, become less important or irrelevant, or decay with time [26].

3.2 Web Services Model

In SOC environments, services from different providers might have to be combined to establish a complete service. For example, consider the composition of three services ($S_1 op_1 S_2$) $op_2 S_3$ as shown in Fig. 3, where S_i 's are services and op_i 's are arbitrary composition operators. Many business domains require composing existing services in order to deliver new functionality. Service composition therefore extends the notion of service discovery by enabling automatic composition of services to meet the requirements of a given a high level task description.

In general, service composition is defined by an open, standards-based approach for connecting services together to create higher-level business processes. Standards are designed to reduce the complexity required to compose services, hence reducing time and costs, and increasing overall efficiency in businesses. Standards for Web services, such as BPEL4WS (Business Process Executable Language for Web Services), enable creating of compositions of Web services as well as defining business processes as coordinated sets of Web service interactions.

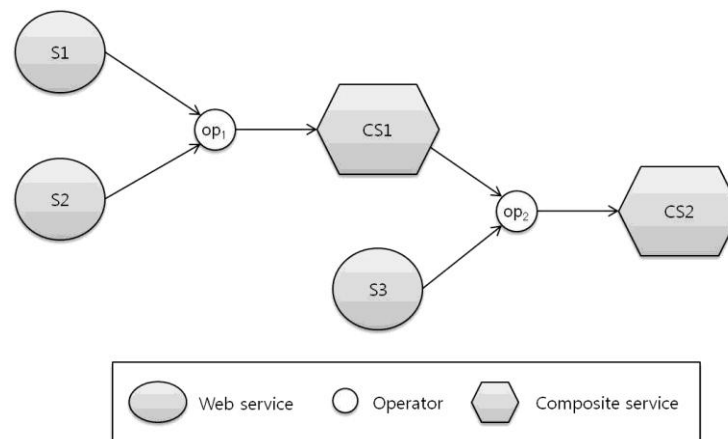


Fig. 3. An example of a composition

When the trust level of each service is known (e.g. untrusted, partially trusted, or fully trusted), what is the trust level of combined service? Suppose that services S_1 and S_2 are only partially trusted, but that service S_3 is fully trusted. Evidently, the service consumer cannot fully trust the composed service. Although the individual service can be trusted, the composed service is not guaranteed to be trustworthy. Thus we need to find a way to determine whether

the composed service can be trusted [27]. To select QoS guaranteed services or credible service providers for composite one as well as individual one, we define a trust model to evaluate the trust level of individual service and deduce the trust level of composed service.

BPEL4WS describes the control logic for Web services coordination in a business process. For example, sequential control between activities is provided by <sequence>, <if>, <while>, <repeatUntil>, and the serial variant of <forEach>. In BPEL4WS, there are four different structures to describe composite services: sequential flow, conditional branching, looping, and parallel processing [28]. To construct a composite service, we consider the execution flow of four structures.

3.3 Trust Model

Traditional approaches model trust qualitatively, based on an intuition of hard security. If one cannot definitely determine that a particular party has the stated identity, then that is sufficient reason not to deal with it at all. Yet, in many cases, requiring an all-or-nothing decision about trust can be too much to ask for, especially when we think not of identity but more broadly of whether a given party would support one's plans.

This paper develops a well-formulated mathematical system with which to approach trust. Intuitively, the trustworthiness of a service should be estimated based on both direct and indirect experience. Direct experience means that there is a previous quality of service that has been received from the service, whereas indirect experience comes from referrals by peers. Thus we model trust using Equation (1). The trust value $\tau_{i,j}(t, c)$ represents the trust degree of an agent a_i toward agent a_j in time t and context c .

$$\tau_{i,j}(t, c) = E_{i,j}(t_0, t_i) \times QoS_{i,j}(t, c) + (1 - E_{i,j}(t_0, t_i)) \times RF_{i,j}(t, c) \quad (1)$$

In Equation (1), $E_{i,j}(t_0, t_i)$ represents the experience factor that indicates the knowledge of agent a_i toward agent a_j that was acquired through direct interaction between a_i and a_j within the time interval $[t_0, t_i]$. We assume that as the number of interactions increases, the knowledge of agent a_i toward agent a_j increases. However, as time lapses, the knowledge level converges to some point. This paper therefore defines the experience factor as a logarithmic growth curve:

$$E_{i,j}(t_0, t_i) = 1 - e^{-k}, \text{ where } k = N_{i,j}/2 \quad (2)$$

In Equation (2), $N_{i,j}$ is the total number of interactions between agents a_i and a_j , and e is a constant approximately equal to 2.7183 as a base of the natural logarithm. Exponential decay models of this form can model learning curves. In the exponential decay model shown in Fig. 4, the experience factor grows rapidly in the beginning, and then levels off to become asymptotic to an upper limit.

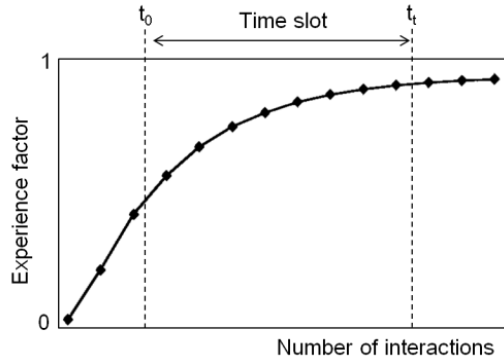


Fig. 4 A learning curve representing the experience factor

In Equation (1), $QoS_{i,j}(t, c)$ and $RF_{i,j}(t, c)$ are as follows:

- $QoS_{i,j}(t, c)$ represents an attribute comes from direct experience. That is a previous quality of service that has been received from the service. For simplicity, we normalize the qualities to the real numbers that lie between 0 and 1. Thus we represent an experience of agent a_i toward a particular quality of a service instance a_j at time t during $[t_0, t_i]$ and within context c as a real number $D_{i,j}(t, c)$ between 0 and 1. Some qualities, availability for example, can be simply considered as 1 (positive) or 0 (negative).
- $RF_{i,j}(t, c)$ represents an attribute comes from indirect experience. That is a referral by peers. We estimate the indirect experience of a service from trust values of agents that interact directly with the service. Thus we also represent the indirect experience of agent a_i toward a service instance a_j at time t during $[t_0, t_i]$ and within context c as a real number $ID_{i,j}(t, c)$ between 0 and 1.

4. Trust Evaluation

4.1 Trust measurement

To evaluate the level of trust, we first assume that the users of the service can assess trustworthiness of the service providers for every transaction. That is, before agent a_i makes interaction with agent a_j , a_i 's trustworthiness towards a_j has to be assessed according to the feedback on the history from the trust system. In order to evaluate and assign the trust level to each service and service provider to select quality-guaranteed services or credible service providers, we consider the concept of social experience formed by direct and indirect participation of users within their own societies. Trust level is considered with direct and indirect experiences. Thus we define confidence from direct experience and the reputation from indirect experience for the service and the service provider.

From Equation (1), for the sake of convenience, the two variables t and c are omitted, as in Equation (3). The trust mediator determines the trust level by using the Equation (3) according to historical data of past interactions. To represent the reputation, we use feedback indicating the level of the customer's satisfaction after use. We assume that the ratings provided by the service consumers are available and are valid.

$$\tau_{i,j} = E_{i,j} \times QoS_{i,j} + (1 - E_{i,j}) \times RF_{i,j} \quad (3)$$

(1) Direct measure

In Equation (3), $QoS_{i,j}$ represents the satisfaction degree of an agent a_i towards an agent a_j according to QoS attributes. This is to directly acquire a satisfaction degree of the service by evaluating the non-functional aspect of each service using several QoS properties.

We choose the quality factors of the service level quality measurements defined by the OASIS standard as QoS attributes: response time, maximum throughput, and availability. Since a service could be provided by third parties and could be invoked dynamically via a network, the service performance might vary depending on the network speed or the number of users that are connected at a given time. Service level quality measurements are a set of quantitative attributes that describe the runtime service responsiveness from the point of view of consumers. This quality factor represents how quickly and soundly Web services can respond and can be measured numerically on a system [29].

Moreover, we also consider the price as another factor. The price of a Web service can be determined by a service provider. A service consumer considers the price of a Web service according to the functions, contents, and the quality of the web service in order to make decisions on whether he uses the web service. Thus the QoS value on each quality attribute defined by [29] is as follows:

- Execution price. The execution price is a monetary value of a service that a consumer pays for such service to a provider during or after using the Web services.
- Response time. It refers to the duration from the time a request is sent to the time a response is received.
- Throughput. It refers to the amount of services that the service provider can process over a given time period. It is the number of responses that can be processed within a unit of time.
- Availability. It is a measurement that represents the degree to which Web services are available in an operational status. This refers to a ratio of time in which the Web services server is up and running.

Let QP, QR, QT and QA stand for price, response time, throughput, and availability, respectively. Only three attributes are considered in this paper, but more attributes can be added depending on the context. For each QoS attribute q_x in $Q = \{QP, QR, QT, QA\}$, a weight is assigned according to its importance in such a way that the following condition holds: for each q_x in Q ($1 \leq x \leq 4$), q_x has the weight w_x where $0 \leq w_x \leq 1$ and $\sum w_x = 1$.

In order to have a uniform measure independent of units, we normalize the QoS values using the method defined in [30]. Liu et al. define a two-pass normalization method for which we adopt only the first normalization step to distribute the computed QoS value over the closed interval $[0, 1]$. Using this normalization, the QoS value provides a uniform index to represent the service qualities for each provider. The provider can increase and decrease his/her quality index by entering a few parameters. Moreover, they can set a threshold regarding those qualities. To compute the QoS value using the above four criteria $Q = \{QP, QR, QT, QA\}$, we can obtain the following matrix QM . Assuming that $S = \{s_1, s_2, \dots, s_n\}$ is a set of Web services that have the same functional properties. Each row in QM represents a Web service s_y , while each column represents one of the QoS criteria q_x in Q .

$$QM = \begin{bmatrix} q_{1,1} & q_{1,2} & q_{1,3} & q_{1,4} \\ q_{2,1} & q_{2,2} & q_{2,3} & q_{2,4} \\ \vdots & \vdots & \vdots & \vdots \\ q_{n,1} & q_{n,2} & q_{n,3} & q_{n,4} \end{bmatrix} \quad (4)$$

To normalize the matrix, we need to define two vectors. The first vector is $N = \{n_1, n_2, n_3, n_4\}$ where the value of $n_j (1 \leq j \leq 4)$ can be 0 or 1. $n_j = 1$ is used in the case where the increase in q_i benefits the service requester while $n_j = 0$ is used for the case where the decrease of q_i benefits the service requester. The second vector is $C = \{c_1, c_2, c_3, c_4\}$. Here $c_j (1 \leq j \leq 4)$ is a constant which sets the maximum normalized value. Each element in vector V will be normalized using Equation (5) and (6).

$$v_{i,j} = \begin{cases} \frac{q_{i,j}}{\frac{1}{n} \sum_{i=1}^n q_{i,j}} & \text{if } \frac{1}{n} \sum_{i=1}^n q_{i,j} \neq 0 \text{ and } n_j = 1 \\ & \text{and } \frac{q_{i,j}}{\frac{1}{n} \sum_{i=1}^n q_{i,j}} < c_j \\ c_j & \text{if } \frac{1}{n} \sum_{i=1}^n q_{i,j} = 0 \text{ and } n_j = 1 \\ & \text{and } \frac{q_{i,j}}{\frac{1}{n} \sum_{i=1}^n q_{i,j}} \geq c_j \end{cases} \quad (5)$$

$$v_{i,j} = \begin{cases} \frac{q_{i,j}}{\frac{1}{n} \sum_{i=1}^n q_{i,j}} & \text{if } q_{i,j} \neq 0 \text{ and } n_j = 0 \\ & \text{and } \frac{q_{i,j}}{\frac{1}{n} \sum_{i=1}^n q_{i,j}} < c_j \\ c_j & \text{if } q_{i,j} = 0 \text{ and } n_j = 0 \\ & \text{and } \frac{q_{i,j}}{\frac{1}{n} \sum_{i=1}^n q_{i,j}} \geq c_j \end{cases} \quad (6)$$

Applying these two equations to QM , we get matrix QM' which is shown below:

$$QM' = \begin{bmatrix} v_{1,1} & v_{1,2} & v_{1,3} & v_{1,4} \\ v_{2,1} & v_{2,2} & v_{2,3} & v_{2,4} \\ \vdots & \vdots & \vdots & \vdots \\ v_{n,1} & v_{n,2} & v_{n,3} & v_{n,4} \end{bmatrix} \quad (7)$$

Then we can finally compute $QoS_{i,j}$ using Equation (8). Through the interaction between agent a_i and a_j , each QoS attribute is measured according to historical data of past interactions, such as logging data, actual execution time, and the number of response times to the user's requests.

$$QoS_{i,j} = \sum (v_{j,k} \times w_k) \quad (8)$$

, where $v_{i,k}$ is the element of QM' and w_k is a weight for the quality value $v_{i,k}$.

(2) Indirect measure

In real situations, people evaluate others based on local views of the world. To form an opinion about an unknown person, we turn to people we know and ask about the unknown person's reputation. In most cases, we do not rely on some centralized authority to determine the reputation each person deserves. One of the fundamental characteristics of indirect measurements is that it is a purely local analysis. To indirectly measure the trustworthiness of a service, we modify the reputation inference algorithm defined in [25]. In this paper, as shown in Fig. 6, our method begins with A's adjacent nodes and expands out to infer a reputation rating of node B or B' by starting at node A.

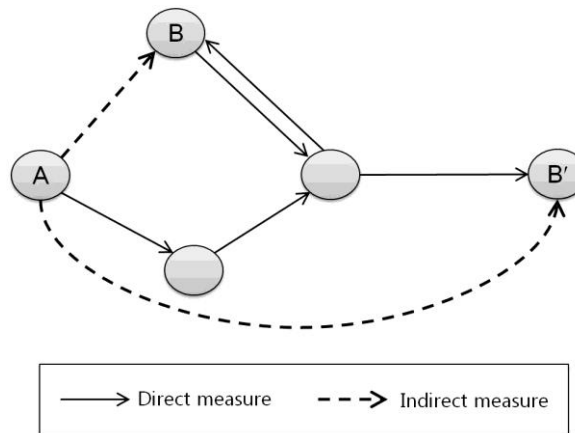


Fig. 3. Indirect measure between two nodes with no interactions

Table 1 shows the algorithm used to compute $RF_{i,j}$ in Equation (3). In the following algorithm, the sink is the node for which a rating is desired and the source is the node for which the rating will be made. That is, the source is the agent a_i and the sink is the agent a_j for $RF_{i,j}$. In this metric, the source polls each of the neighbors to which it has given a trust rating. Each of the source's adjacent nodes will return their rating for the sink. The source will then average these ratings. The mean value is the inferred trust rating from source to sink, and the function $Rating(a, b)$ gets a trust value $QoS_{a,b}$ from direct measures between agents a and b .

Table 1. The computation algorithm for $RF_{i,j}$

```

Input: source and sink
Output: the trust value of the sink
TrustInfer(source, sink) {
    numberOfAdjacentNodesWithRatings = 0 ;
    sumOfRatings = 0 ;
    mark source seen ;
    if sink is adjacent to source
        source's rating of sink = Rating(source, sink) ;
    else
    {
        for each n adjacent to source {
            if n is unseen {
                if n has no ratingOfSink {
                    mark n seen ;
                    inferredRating = TrustInfer(n, sink) ;
                    sumOfRatings += inferredRating ;
                }
            }
        }
    }
}

```

```

        n's rating of sink = inferredRating ;
        numberOfAdjacentNodesWithRatings ++ ;
        mark n unseen;
    }
    else {
        sumOfRatings += n's rating of sink ;
        numberOfAdjacentNodesWithRatings ++ ;
    }
} //end for
if numberOfAdjacentNodesWithRatings > 0
    source's rating of sink =
        sumOfRatings/numberOfAdjacentNodesWithRatings ;
else
source's rating of sink = ∞ ;
} //end else
return source's rating of sink ;
} //end TrustInfer( )

```

Each interaction between agents has to accumulate information with respect to the sequence of events that has occurred. In addition, every agent should have its own trust policy to check for compliance with trust-related requirements. Declarative policy languages, such as WS-Policy, are used to express the capabilities and requirements of the entities in a Web-service-based system.

4.2 Trust estimation for composing services

Web service composition is a compilation of several services aggregated to execute in sequence. A complex composition problem with a parallel implementation, branching, and loops is usually an NP-hard problem. To select a service, we have a set of candidate services. Depending on the user need, non-functional aspects can be selected.

In [5], Hang and Singh give some example on how some quality metrics are composed in the type of interactions. They propose quality metrics such as SWITCH, MAX, MIN, SUM and PRODUCT. For example SUM yields the composite quality value as the sum of the quality values obtained from all constituent services. However, the highest total sum of the quality value cannot guarantee that the composite service is the most trustworthy because the summation depends on the number of constituent services.

The trustworthiness of a composite service is computed via propagation of the trust values of atomic services. Here three strategies can be utilized according to Kuter and Golbeck [31]: *Overly-Cautious*, *Overly-Optimistic* and *Average*. All these strategies aim to find a composition with the highest trust value. The goal of the first strategy is to maximize the minimum expected trust value that the composer user has in the atomic services of the composite service. In other words, it assumes that if something bad could happen, it would definitely happen, and thus it avoids incorporating low-trust services. In contrast to the first strategy, an *Overly Optimistic* strategy promotes the influence of highly trusted atomic services into trust of the composite service because it believes that nothing bad happens if low-trust services are taken into account for composition. The last strategy is an intermediate approach that looks for compositions with a maximum average trust.

We adopt an *Overly-Cautious* strategy for defining the trust in a service S that aims to maximize the minimum expected trust value that the composer user has in the sub-processes of

S . Consider composition operators that integrate services to build a business process. As described in Section 3.3, there are four structures that can be used to compose services: sequential flow, looping, conditional branching, and parallel processing. Let OP be the set of composition operators for describing four structures. Let S be a composite service and $\{s_1, s_2, \dots, s_n\}$ be a set of available atomic services to construct the composite service S . We denote $S = \{s_1, s_2, \dots, s_n\}$. Each atomic service s_i in S has the trust value $t(s_i)$ of s_i computed by the average of the trust values on the incoming edges into node s_i .

$$t(s_i) = \frac{1}{\alpha} \sum_{k=1}^{\alpha} \tau_{k,i} \quad (9)$$

In Equation (9), $\tau_{k,i}$ is the trust value of agent a_k toward agent s_i by Equation (3), where a_k is an adjacent node of s_i . Then S is defined by business process models and atomic services are executed on the business process model. Let $BP(S) = \{bp_1, bp_2, \dots, bp_m\}$ be a set of business process models used to construct the composite service S . Each bp_j is defined by a tuple $bp_j = (OP', S')$ where $OP' \subseteq OP$ and $S' \subseteq S$. In formal, a trust value $t(S)$ for S is denoted as in the following Equation (12):

$$t(S) = \max_{bp_j \in BP} Q(bp_j), \quad Q(bp_j) = \min_{s \in S'} t(s_i) \quad (10)$$

For example, consider the process model BP , as shown in Fig. 6, to build a composite service $S = \{s_1, s_2, \dots, s_7\}$ based on business processes $BP = \{bp_1, bp_2, bp_3\}$. For each service s_i in S , $t(s_i)$ is the trust value of s_i . Assume that $(t(s_1), t(s_2), \dots, t(s_7)) = (0.78, 0.25, 0.34, 0.36, 0.52, 0.69, 0.83)$. Then $Q(bp_1) = \min(0.78, 0.25, 0.36, 0.52, 0.69) = 0.25$, $Q(bp_2) = \min(0.34, 0.36, 0.52, 0.69, 0.83) = 0.34$, and $Q(bp_3) = \min(0.78, 0.36, 0.52, 0.69, 0.83) = 0.36$. The final trust value of the composite service S is 0.36 of $\max(bp_1, bp_2, bp_3)$.

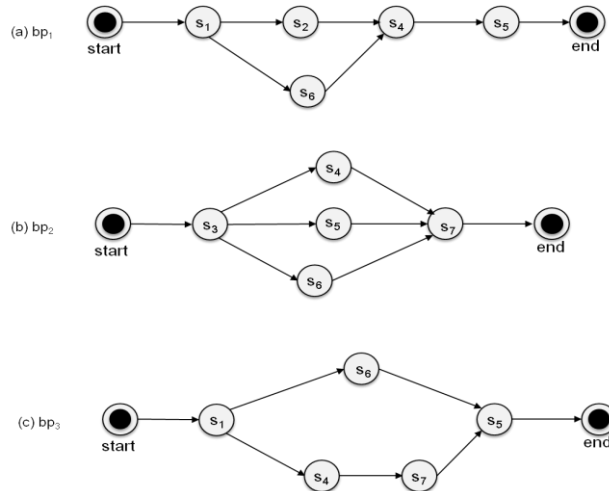


Fig. 4. An example of the service composition

5. Experimental evaluation

We performed experiments to evaluate the models presented in this paper. First of all, Fig. 7 shows the framework used to implement the experiments. Our model consists of the trust manager, composition engine, and QoS analyzer. Once a service provider enrolls service descriptions in the UDDI service registry, the trust manager constructs trust information of the service and the service providers. Each time the service is evaluated through service monitoring, the trust manager updates trust information of the service. After assigning the service's trust

information, the trust manager modifies the past service profile with respect to the trust level. The QoS analyzer should have event structures that maintain transactions during specific time-intervals. Changes to the trust level of the services in a group have no effect on other service groups. This manages trust information efficiently. To implement the trust model, we need away to describe the QoS attributes of a service. The QoS of the service can be attached to the *ServiceProfile* in OWL-S, a new class as a subclass of the *ServiceParameter* class already defined in OWL-S. **Table 2** is an example of QoS requirements described in OWL-S.

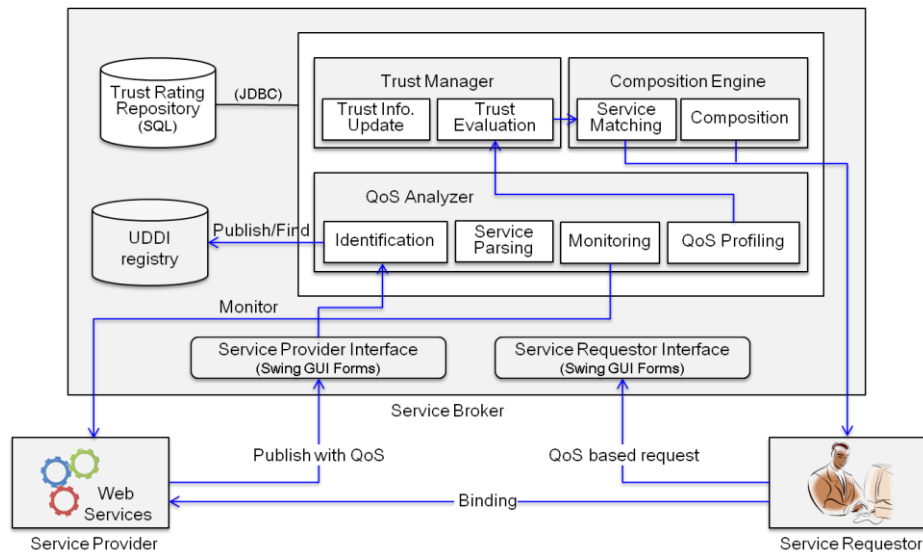


Fig. 5. Trust evaluation model prototype

We have conducted two experiments running on a framework implemented on jUDDIv3 and MySQL. We collect 120 services from e-commerce systems. The services are classified into three logical groups providing similar functionalities with different quality levels. Registered services have functional capabilities and QoS information.

Table 2. OWL-S description of service requirements

```

<Service Quality rdf:ID="ServiceQuality_1">
  <profile:sParameter>
    <ServiceQualityInfo rdf:ID="ServiceQualityInfo_11">
      <ServicePrice rdf:datatype="http://.../XMLSchema#float">200
    </ServicePrice>
    <ResponseTime rdf:datatype="http://.../XMLSchema#float"> 0.03
    </ResponseTime >
    <Availability rdf:datatype="http://.../XMLSchema#float"> 99.5
    </Availability >
    <Throughput rdf:datatype="http://.../XMLSchema#integer"> 700
    </Throughput >
    </ServiceQualityInfo>
  </profile:sParameter>
  <profile:serviceParameterName rdf:datatype="http://.../XMLSchema#string"> ServiceQuality
  </profile:serviceParameterName>
</ServiceQuality>
<profile:Profile rdf:ID="serviceUser1_Profile">
  <profile:serviceParameter rdf:resource"#ServiceQuality_1">
  ...
</profile:Profile>

```

The goal of the first experiment is to verify the fact that the proposed trust evaluation method provides the optimal trusted service that fulfills the user's requirements. Each service is invoked 50 times with different QoS requirements, and at the end of each interaction, the system updates the corresponding trust graph. To show the effectiveness of our approach, we compare the proposed method with the average QoS values in all invocations. We compare the service having the highest value of trust level using Equation (3) with the selected service based on QoS information.

For example of computation of the trust value, **Table 3** shows the values of each quality attribute for arbitrary five services satisfying the requirements in **Table 2** among the collected 120 services. The experiment executes on four quality attributes: price (10,000 won per service), response time (millisecond), throughput (the number of requests completed in second), and availability (%). The five services are in the candidates. As a result of the trust evaluation using Equation (3), the service having the highest trust level is returned.

Table 3. The value of quality attributes

Attributes \ Services	S1	S2	S3	S4	S5
Price(P)	150	100	125	130	110
Response time (RT)	0.01	0.03	0.02	0.03	0.02
Throughput (T)	730	700	720	710	720
Availability (A)	99.8	99.5	99.6	99.7	99.7

For each service and its adjacent nodes, the computation of the direct measure is the first. Note that the trust level of each service S1, S2, S3, S4, and S5 is evaluated by the adjacent nodes. If a service S1 is adjacent with S7, S12, and S20, then the experience factor $E_{7,1}$, $E_{12,1}$, and $E_{20,1}$ and the direct measure $QoS_{7,1}$, $QoS_{12,1}$, and $QoS_{20,1}$ are computed. For example, when S1 and S7 are interacted with each other in 5 times, $E_{7,1} = 1 - e^{-2.5}$. For the convenience of explanation, if $S = \{S1, S2, S3, S4\}$ is a set of adjacent nodes to S7, QM is defined as the following and is normalized in [0, 1] according to Equation (5) and (6) as the following QM'. Then $QoS_{i,j}$ is computed by weighted summation of elements in the j-th row of QM'. That is, $QoS_{7,1} = (1 \times 0.25) + (1 \times 0.25) + (1 \times 0.25) + (1 \times 0.25) = 1$ where the weight is even.

$$QM = \begin{bmatrix} 150 & 0.01 & 730 & 99.8 \\ 100 & 0.03 & 700 & 99.5 \\ 125 & 0.02 & 720 & 99.6 \\ 130 & 0.03 & 710 & 99.7 \end{bmatrix} \quad QM' = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0.67 & 0.33 \\ 0.6 & 0 & 0.33 & 0.67 \end{bmatrix}$$

For the services S2, S3, S4, and S5, the computation of the direct measure is computed similarly. If two nodes S1 and S7 are not adjacent with each other, we have to compute the indirect measure. The algorithm in **Table 1** returns the value of $RF_{7,1}$ as the indirect measure. Then we get the trust values $\tau_{7,1}$, $\tau_{12,1}$, and $\tau_{20,1}$. Consequently, the trust level of the service S1 averages those trust values. **Table 4** shows the trust level for the services in **Table 3**.

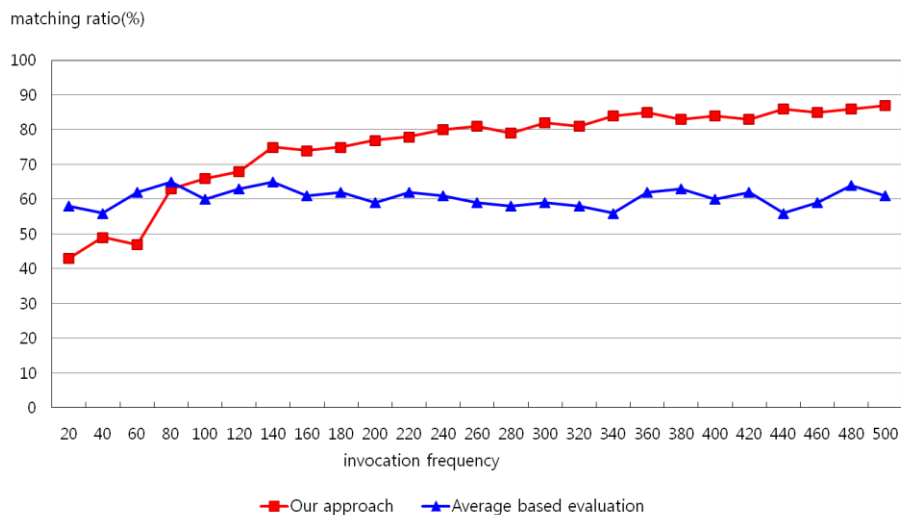
Table 4. The computation result of the trust level

Attribute weight	Trust level				
	S1	S2	S3	S4	S5
Even	0.67	0.65	0.72	0.76	0.81
P=0.4, RT=T=A=0.2	0.6	0.79	0.7	0.62	0.8
RT=0.4, P=T=A=0.2	0.78	0.69	0.73	0.65	0.78

P: Price, RT: Response Time, T: Throughput, A: Availability

Assume that the services that have an appropriate quality level for the QoS attributes are selected first. Then some of them are chosen to evaluate the reputation because the reputation is the only factor for selecting services without trustworthiness. Upon the submission of a user's requests, the service candidates that satisfy functional and quality attributes for the user's requirements are selected. Among the service candidates, only the service with the highest trust level is returned. In **Table 4**, the service S5 remains the most trustworthy service independent with the weight of quality attributes. Thus the proposed method returns always the service S5 as a trustworthy service fulfilling the user's requirements. However the experimental results do not come to the conclusion that the attribute weight is independent with the trust level. It is needed to execute further refined experiments with several weight variations.

Fig. 8 shows the results of our first experiment where the ratio of the right selection of two approaches is much the same as when the number of times of the invocation is less than 100. Since our trust model is based on a direct interaction, the proposed model is helpful when experience data is sufficient after a certain amount of time has lapsed. The right selection ration on average of our approach is 83.63%, and that of the average based approach is 71.58%.

**Fig. 6.** The correct selection ratio

In the second experiment, the composite service "Purchase Order" is chosen as an experimental scenario. It is based on the "Purchase Order Process" example taken from the BPEL 2.0 specification as shown in **Fig. 9**. The WSDL service descriptions of the example are used to define the composite service "Purchase Order"[32]. To compose the service, we

implement forty candidate services with differentiated QoS values. The goal of this experiment is to show whether the proposed trust evaluation method also helps to estimate the trust of the composite service based on the user’s QoS requirements.

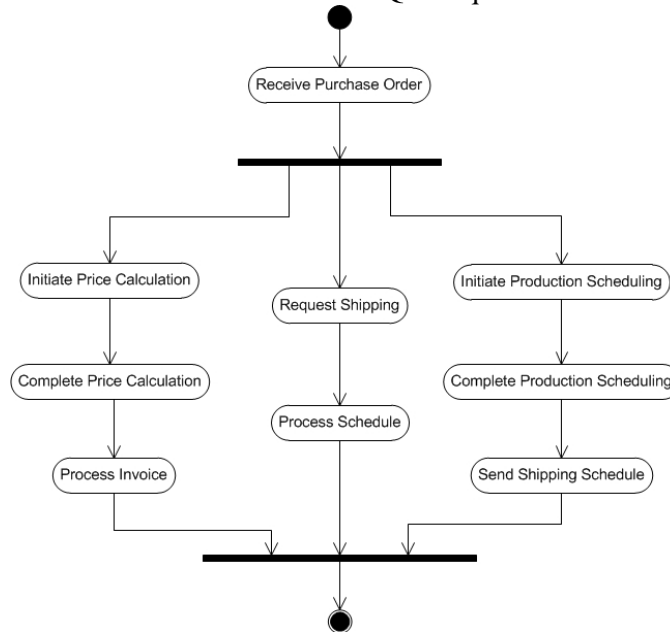


Fig. 7. The Purchase Order Process Outline [32]

To show the efficacy of our approach, we compare the proposed method against a random walk in all compositions. To construct a composite service, our approach selects constituent services using a trust based min-max strategy described in the Section 4.2 while the random walk selects constituent services arbitrarily from candidate services. We measure the success rate, which is the ratio of the operation sequences that are successfully executed by the composite service. In our experiments, to measure the success rates, we generated 10 business process models as defined by different candidate services. For each business process model, all possible operation sequences are generated by composition operators.

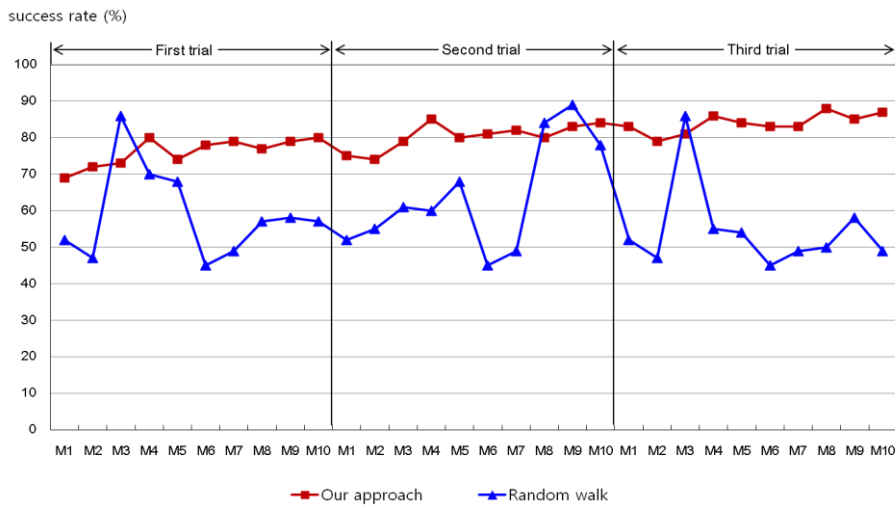


Fig. 8. The success rate of the composition

Fig. 10 shows the results of our second experiment, indicating the success rates of the execution of composite services. In **Fig. 10**, the horizontal axis represents 10 business process models with three trials. For thirty total operation sequences, the success rate on average of our approach is 80.1%, and that of the random walk is 59.1%. Our approach can provide trustworthy composite services stably by estimating the trustworthiness of the service as compared against a random approach. Since our trust evaluation of a composite service is based on the average of the trust values acquired from direct and indirect experiences, it may guarantee that we select more trustworthy elementary services to construct a composite service. Consequently, the composite service consisting of trustworthy elementary services can be trusted.

6. Conclusion

Recently, SOA has emerged as a very popular architecture paradigm that can be used to design and develop modern distributed systems, such as cloud computing systems. In distributed, heterogeneous Web services environments, the potential number of services that provide similar functionalities in enterprise software can be extremely large. To provide consumers with useful services, the environment in which services exist should be steady, and all the services in it should be trustworthy. Otherwise, consumers may find that, although the descriptions of services match their requirements perfectly, the services they select may not be the most trusted. In critical fields, such as medical or financial systems, poorly-informed decisions can have unacceptable negative consequences.

To create more complex, value-added, cross-organizational business processes, single services are combined into composite services. In this case, although an individual service can be trusted, the composed service is not guaranteed to be trustworthy. A consumer may interact with a composite service without knowing much about the qualities of the underlying services. In such a case, evaluating the trustworthiness of a service is nontrivial. Thus, the service selection should consider service compositions to model how the quality of a component service can affect the whole composition. Thus a way to efficiently select and compose trusted services is needed. By facilitating QoS-driven discovery and composition, users can easily and efficiently select trustworthy services that are most suitable to their needs.

In this paper, we present a trust model that supports discovery and composition of trustworthy services based on QoS properties. The model includes an evaluation method based on quality attributes observable from direct and indirect experiences of a service consumer and the estimation method to provide the trust value of the composite service. The model prototype is implemented on a QoS-broker based architecture for Web services, and we have conducted two experiments running on the framework. The goal of the first experiment is to verify the fact that the proposed trust evaluation method provides the optimal trusted service that can fulfill the user's requirements. The result for the ratio of the right selection of two approaches is much the same the number of times of invocation is less than 100. The experiment shows that the proposed model is helpful when there is enough data that has been collected as time lapsed because our trust model is based on direct interactions. The goal of the second experiment is to show whether the proposed trust evaluation method helps to also estimate the trust of the composite service based on the user's QoS requirements. The result is that the success rate of our approach is 80.1%, and that of a random walk is 59.1%, on the average, for thirty operation sequences in total. Our approach can provide a trustworthy composite service stably by estimating the trustworthiness of the service as compared against the random approach.

Existing research on trust leaves open significant research and technological issues which, from a software service perspective, are related to: (1) the assessment of trust in a dynamic deployment and the composition of software services, (2) the evaluation of the accuracy and the risk of trust assessments, (3) acquisition of trust information due to deviations of services from normal behavior in a wide range of contexts, (4) the evaluation of the context relevance for trust assessments, (5) the bootstrapping of trust assessment environments, and (6) the provision of an interoperable, robust and secure runtime platform that can realize independent trust certification services. Our research will therefore be further refined to resolve these issues. Lastly, QoS based service composition will be adapted in the real world safe and sound provided that QoS integrity is preserved at all service providers, and the OWL-S execution platform that supports the dynamic service grounding is deserved to be developed. These problems remain as future research topics.

References

- [1] P. Bianco, R. Kotermanski, and P. Merson, "Evaluating a Service-Oriented Architecture," *Technical Report*, Software Engineering Institut, CMU/SEI-2007-TR-015, 2007.
[Article \(CrossRef Link\)](#)
- [2] Z. M. Aljazzaf, *Trust-Based Service Selection*, Ph.D. Thesis, Dept. of Computer Science, The University of Western Ontario, Ontario, Canada, 2011. [Article \(CrossRef Link\)](#)
- [3] E. Chang, T. Dillon, and F. K. Hussain, *Trust and Reputation for Service-Oriented Environments*, John Wiley & Sons, Inc., 2006.
- [4] D. Gambetta, *Trust: making and breaking cooperative relations*, Basil Blackwell Ltd., 1988.
[Article \(CrossRef Link\)](#)
- [5] C. W. Hang and M. P. Singh, "Trustworthy Service Selection and Composition," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 5, no. 4, pp. 1–18, 2010.
[Article \(CrossRef Link\)](#)
- [6] N. Dragoni, "Toward trustworthy web services - approaches, weaknesses and trust-by-contract framework," in *Proc. of the International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT'09)*, pp. 599–606, 2009. [Article \(CrossRef Link\)](#)
- [7] W. Shao-Jie, S. Gui-Cheng, Z. Xue-Feng, C. Li-Jun, and Y. Zhen, "A trust model of web services based on individual experience," in *Proc. of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCom'07)*, pp. 3205–3208, 2007.
[Article \(CrossRef Link\)](#)
- [8] G. Wu, J. Wei, X. Qiao, and L. Li, "A Bayesian network based QoS assessment model for web services," in *Proc. of the International Conference on Services Computing (SCC'07)*, pp. 498–505, Salt Lake City, 2007. [Article \(CrossRef Link\)](#)
- [9] W. Lin, C. Lo, K. Chao, and M. Younas, "Consumer-centric QoS-aware selection of web services," *Journal of Computer and System Sciences*, vol. 74, no. 2, pp. 211–231, 2008.
[Article \(CrossRef Link\)](#)
- [10] K. Yue, W. Liu, and W. Li, "Towards web services composition based on the mining and reasoning of their causal relationships," *Lecture Notes in Computer Science*, vol. 4505, pp. 777–784, 2007. [Article \(CrossRef Link\)](#)
- [11] S. Paradesi, P. Doshi, and S. Swaika, "Integrating behavioral trust in web service compositions," in *Proc. of the International Conference on Web Services (ICWS'09)*, pp. 453–460, 2009.
[Article \(CrossRef Link\)](#)
- [12] Y. Wang, and M. Singh, "Evidence-based trust: A mathematical model geared for multiagent systems," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 5, no. 4, 2010.
[Article \(CrossRef Link\)](#)
- [13] E. Maximilien, and M. Singh, "A framework and ontology for dynamic web services selection," *IEEE Internet Computing*, vol. 8, no. 5, pp. 84–93, 2004.

- [Article \(CrossRef Link\)](#)
- [14] D. Artz, and Y. Gil, "A Survey of Trust in Computer Science and the Semantic Web," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 2, pp.58-71, 2007. [Article \(CrossRef Link\)](#)
- [15] M. Fernandez-Gago, R. Roman, and J. Lopez, "A survey on the Applicability of Trust Management Systems for Wireless Sensor Networks," in *Proc. of the International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SECPeU'07)*, pp.25-30, 2007. [Article \(CrossRef Link\)](#)
- [16] S. Marti, and H. Garcia-Molina, "Taxonomy of Trust: Categorizing P2P Reputation Systems," *Computer Networks*, vol. 50, no. 4, pp. 472-484, 2006. [Article \(CrossRef Link\)](#)
- [17] A. Josang, R. Ismail, and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618-644, 2007. [Article \(CrossRef Link\)](#)
- [18] G. Silaghi, A. Arenas, and L. Silva, "Reputation-based Trust Management Systems and Their Applicability to Grids," Technical Report. [Article \(CrossRef Link\)](#)
- [19] Y. Wang, and J. Vassileva, "Toward Trust and Reputation Based Web Service Selection:A Survey," *International Transactions on Systems Science and Applications*, vol. 3, no. 2, pp. 118-132, 2007. [Article \(CrossRef Link\)](#)
- [20] [20] T. Zhang, J. Ma, C. Sun, Q. Li and N. Xi, "Service Composition in Multi-Domain Environment under Time Constraint," in *Proc. Of the International Conference on Web Services (ICWS2013)*, pp. 227-234, 2013. [Article \(CrossRef Link\)](#)
- [21] S. Bansal, A. Bansal and M. B. Blake, "Trust-based Dynamic Web Service Composition using Social Network Analysis," in *Proc. of International Workshop on Business Applications of Social Network Analysis (BASNA 2010)*, pp. 1-8, 2010. [Article \(CrossRef Link\)](#)
- [22] Y. Wang, I. Chen, J. Cho, K. Chan and A. Swami, "Trust-based Service Composition and Binding for Tactical Networks with Multiple Objectives," in *Proc. of Military Communications Conference (MILCOM'13)*, pp. 1862-867, 2013. [Article \(CrossRef Link\)](#)
- [23] X. Wu, B. Li, R. Song, C. Liu and S. Qi, "Trust-based Service Composition and Optimization," in *Proc. of Asia-Pacific Software Engineering Conference (APSEC 2012)*, pp. 67-72, 2012. [Article \(CrossRef Link\)](#)
- [24] Ws-Trust 1.4. OASIS Standard. [Article \(CrossRef Link\)](#)
- [25] J. Golbeck, *Computing and Applying Trust in Web-based Social Networks*, Ph.D. Thesis, 2005. [Article \(CrossRef Link\)](#)
- [26] T. Grandison and S. Sloman, "A survey of trust in Internet applications," *IEEE Communications Surveys and Tutorials*, vol. 3, no. 4, pp. 2-16, 2000. [Article \(CrossRef Link\)](#)
- [27] Y. Kim, Y. Shin and K. G. Doh, "Quantitative Trust Management with QoS-aware Service Selection", *International Journal of Web and Grid Services*, Inderscience Enterprises Ltd. (accepted).
- [28] WS-BPEL Version 2.0 Primer, OASIS standard. [Article \(CrossRef Link\)](#)
- [29] OASIS Standard, Web Services Quality Factors Version 1.0. [Article \(CrossRef Link\)](#)
- [30] Y. Liu, A. Ngu and L. Zeng, "QoS Computation and Policing in Dynamic Web Service Selection," in *Proc. of the International World Wide Web Conference (WWW2004)*, pp. 66-73, 2004. [Article \(CrossRef Link\)](#)
- [31] U. Kuter, J. Golbeck, "Semantic Web Service Composition in Social Environments," in *Proc. of the International Semantic Web Conference (ISWC'09)*, pp. 344-358, 2009. [Article \(CrossRef Link\)](#)
- [32] OASIS Standard, Web Services Business Process Execution Language (v2.0), [Article \(CrossRef Link\)](#)



Yukyong Kim is a Research Professor of Dept. of Computer Science and Engineering at Soongsil University. She received her M.S. and Ph.D. in Computer Science from Sookmyung Women's University, Seoul, Korea in 1994 and 2001, respectively. She received her B.S. in Mathematics and Computer Science from Sookmyung Women's University, Seoul, Korea in 1991. She was a post-doctoral researcher in the Dept. of Computer Science at the University of California at Davis, USA in 2005 - 2006. Her research interests include service-oriented architecture, quality of services, and software quality assurance.



Jong-Seok Choi is a Ph.D. Student of Dept. of Computer Science and Engineering at Soongsil University, Seoul Korea. He received his B.S. and M.S. in Computer Science from Soongsil University, Seoul, Korea in 2012 and 2014, respectively. His research interests are network security, computer communications, and personal information protection.



Yongtae Shin is a Professor of Dept. of Computer Science and Engineering, and Dean of the graduate school of software at Soongsil University, Korea. He received his M.S. and Ph.D. in Computer Science from the University of Iowa, USA in 1990 and 1994, respectively. He received his B.S. in Industrial Engineering from Hanyang University, Seoul, Korea in 1985. His research interests focus on Multicast, IoT, Information Security, Content Security, Mobile Internet, Next Generation Internet. He was a director of National IT Industry Promotion Agency (NIPA) and Korea Internet Security Agency. He served as the vice president of the Korean Institute of Information Scientists and Engineers, Korea Information Processing Society, and the Korean Society of Internet Ethics.