Integrating Behavioral Trust in Web Service Compositions

Sharon Paradesi, Prashant Doshi *and* Sonu Swaika
LSDIS Lab, Dept. of Computer Science
University of Georgia
Athens, GA, 30602
{paradesi,pdoshi,swaika}@cs.uga.edu

Abstract

Algorithms for composing Web services (WS) traditionally utilize the functional and quality-of-service parameters of candidate services to decide which services to include in the composition. Users often have differing experiences with a WS. While trust in a WS is multi-faceted and consists of security and behavioral aspects, our focus in this paper is on the latter. We adopt a formal model for trust in a WS, which meets many of our intuitions about trustworthy WSs. We hypothesize predictors of a positive experience with a WS and conduct a small pilot study to explore correlations between subjects' experiences with WSs in a composition and the predictor values for those WSs. Furthermore, we show how we may derive *trust for compositions from trust models of individual services. We conclude by presenting and evaluating a novel framework, called* Wisp*, that utilizes the trust models and, in combination with any WS composition tool, chooses compositions to deploy that are deemed most trustworthy.*

1 Introduction

Web service composition (WSC) techniques traditionally utilize the functional and quality-of-service (QoS) parameters of candidate services to decide which services to include in the composition [5]. However, users of services often form an opinion, somewhat subjective, of a service. This opinion may be based on prior interactions with the service, and may include judgments such as whether the perceived behavior of the service conforms to its stated behavior and intangibles such as the overall experience of the user with the service. WSC techniques that additionally consider this assessment of services by users will form compositions that likely behave in practice as stated, and which are better received by the users.

Trust in a WS is multifaceted and includes a security based aspect such as establishing the authenticity and authority of the service, and behavioral (social) aspect such as judging whether the service behaves reliably and as advertised. The focus of this paper is on studying the behavioral aspect of trust. We begin by mathematically formalizing a notion of trust in a service. We may consider a WS to be trustworthy if we strongly believe that the probability of having a positive experience with the WS is high. On the other hand, we assign a low belief to high probabilities of a positive experience with a WS deemed untrustworthy. Given that belief is often formalized as a probability distribution, trust may be modeled using a *second order probability density function* (sometimes called a probability certainty density function (PCDF) [9]). This stochastic model for trust follows the model of Wang and Singh [16] which targets general agents, and which is itself a modification of Josang's [9].

We hypothesize that a *positive experience* of users with a WS correlates with the perceived honesty, reliability and competency of the WS. We may objectively measure these variables of a service participating in a WSC, thereby allowing us to test any hidden correlation between users' subjective opinions and objective measurements. In this context, we conduct a *pilot study* mapping the experiences of users interacting with services in example WSCs to the objective measures of the three aspects for individual WSs. Data from this study is used in testing our hypothesis and in formulating different user preference models, to be used in later experimentation.

Following Wang and Singh [16], belief over individual WSs is progressively updated based on user experiences . Although trust could be treated as a QoS parameter, this would require that existing WSC techniques be adapted to account for trust while forming the composition. To avoid this, we develop a separate trust framework, which we call Wisp, that computes the aggregate trust in the composition, and selects the composition that is most trustworthy. This is additionally useful because the composed WS may itself be used as a component service later. Furthermore, we present a method for comparing the different WSCs based on their derived trustworthiness and report on supporting experimentation.

2 Model of Trust in Web Services

Wang and Singh [16] present a formal model of trust for multiagent systems. It shares its conceptual underpinnings

978-0-7695-3709-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICWS.2009.106

with the formalization of trust proposed by Josang [9], and improves on some of its limitations. We adapt Wang and Singh's general trust model to the specific context of WSs.

2.1 Belief Density

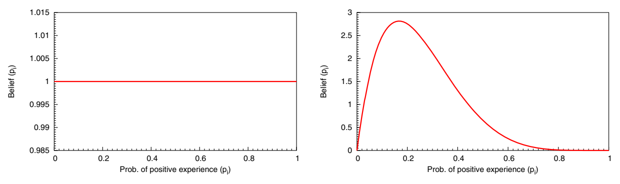
Trust is inherently uncertain, and in simple terms, is derived from the *belief* that a subject has in another's abilities to perform the actions on which the subject's welfare depends, while being cognizant of possible negative consequences. We define our model of belief more formally:

Definition 1 (Belief)**.** *Let $p_i \in [0,1]$ be the probability that users will have a positive experience with WS i. Then, define the* belief *in a positive experience with WS i as a probability density function over p_i, denoted as $B(p_i)$.*

Belief is a density function (and not a discrete distribution) because the space of p_i is continuous. Furthermore, the belief is a *second order* probability function – probability density over a probability; Josang [9] refers to it as a probability certainty density function (PCDF). We ground what could be meant by a 'positive experience' later in this paper.

Initially, we may have no opinion about the behavior of WS i and the belief will be a flat line indicating no information about p_i. As the positive or negative experiences with the WS accumulate, the belief function will assume forms that assign large probability masses to high or low p_i values, respectively.

Example 1. *We show example beliefs in Fig. 1. The belief density in Fig. 1(a) models a state of complete uncertainty about the probability of a positive experience with WS i. On the other hand, Fig. 1(b) indicates that the chances of having a positive experience with the WS i are likely to be low.*



(a) (b)

Figure 1. Belief densities represented using the beta density function. Beta densities are parameterized by a and b. (a) Flat line indicates no information about p_i ($a = b = 1$). (b) Larger probability mass is assigned to low p_i ($a = 2, b = 6$).

While the belief densities could take any shape, we restrict the form to the family of *beta* probability densities. This is beneficial because beta densities are well studied and exhibit the desirable property that they represent a conjugate family of distributions in the context of Bayesian updating:

$$B(p_i; a, b) \overset{def}{=} \frac{1}{\beta(a,b)} p_i^{a-1}(1-p_i)^{b-1} \qquad (1)$$

where a, b parameterize the density and $\beta(a,b)$ is the well-known beta function.

2.2 Belief Update

An important property of trust is that subjects often 'come around' to trusting another person or entity. In other words, trust in a WS *must be updated using past experiences* – both positive and negative. As trust is derived from the belief, $B(p_i; a, b)$, we must develop a way to update the belief with past experiences of users with the WS i.

After n interactions with WS i, let there be r positive experiences and s non-positive (possibly negative) experiences, where $n = r + s$. We wish to update the prior belief, $B(p_i; a, b)$, with these experiences resulting in an updated belief, $B'(p_i|\langle r, s\rangle)$. Notice that the probability of a positive experience is p_i, and that of a non-positive experience is $(1-p_i)$. Given the experiences, we may update the belief over p_i in a Bayesian manner:

$$\begin{aligned} B'(p_i|\langle r,s\rangle) &= \frac{Pr(\langle r,s\rangle|p_i)B(p_i;a,b)}{\int_0^1 Pr(\langle r,s\rangle|p_i)B(p_i;a,b)dp_i} \\ &= \frac{p_i^r(1-p_i)^s B(p_i;a,b)}{\int_0^1 Pr(\langle r,s\rangle|p_i)B(p_i;a,b)dp_i} \\ &= \alpha p_i^r(1-p_i)^s B(p_i;a,b) \\ &= \alpha \frac{1}{\beta(a,b)} p_i^r (1-p_i)^s p_i^{a-1}(1-p_i)^{b-1} \quad \text{(Eq. 1)} \\ &= \alpha' p_i^{(a+r)-1}(1-p_i)^{(b+s)-1} \end{aligned}$$

where α is the normalization constant and $\alpha' = \frac{\alpha}{\beta(a,b)}$. Consequently, the updated belief, $B'(p_i|\langle r,s\rangle)$ remains a beta density function and is parameterized by $a + r$, $b + s$.

2.3 Certainty and Trust Vector

Trust in a WS i represents the level of certainty of the belief in a positive experience with i. While we may measure the Shannon entropy [8] of the belief, it does not provide a measure of the certainty of the second order belief over p_i, an observation also made by Wang and Singh [16].

One way to measure the certainty level of a belief, $B(p_i; a, b)$, is to ascertain how much $B(p_i; a, b)$ is further away from a state of complete uncertainty over p_i. The latter is shown in Fig. 1(a), and could be represented as, $B(p_i; 1, 1) = 1$. We denote this special uniform density as B_u. Following Wang and Singh [16], we utilize the L_1 norm to measure the distance between the densities, although other (pseudo-)distance measures such as the KL Divergence may also be used. We define the certainty below:

Definition 2 (Certainty)**.** *Certainty of a belief over the probability of positive experience with a WS i, $C_i : B \rightarrow \mathbb{R}$, is:*

$$\begin{aligned} C_i(B) &\overset{def}{=} \tfrac{1}{2} L_1\,(B - B_u) \\ &= \tfrac{1}{2}\int_0^1 |B(p_i;a,b) - 1|dp_i \\ &= \tfrac{1}{2\,\beta(a,b)}\int_0^1 |p_i^{a-1}(1-p_i)^{b-1} - 1|dp_i \end{aligned} \qquad (2)$$

Normally, the L_1 norm accumulates the difference between a peak in $B(p_i; a, b)$ and the flat line and a dip in $B(p_i; a, b)$ and the flat line. To avoid counting the difference twice, we scale the L_1 norm of the difference between B and B_u with 0.5. This has the beneficial effect of keeping $0 \le C_i(B) \le 1$.

Example 2. *The certainty of the belief density in Fig. 1(b) is computed as:* $C_i(B') = \frac{1}{2}\int_0^1 |B(p_i; 2, 6) - 1| dp_i = 0.461$

Both trust and distrust could contribute toward the certainty level of the belief over p_i. Hence, what remains is to find a way to distribute the certainty among trust and distrust. Because trust in WS i is an outcome of the positive experiences with i, while distrust results from the non-positive experiences with i, we may assign a proportion of certainty equal to the proportion of positive experiences among the total experiences, to trust. Remaining portion of the certainty is distrust. We introduce the *trust vector* and define it formally:

Definition 3 (Trust vector)**.** *Define the trust vector for WS i as, $\bar{v}_i = \langle t_i, d_i, u_i \rangle$, where $t_i + d_i + u_i = 1$, and,*

$$t_i = \frac{r}{r+s} C_i(B);\; d_i = \frac{s}{r+s} C_i(B);\; u_i = (1 - (t_i + d_i))$$

For simplicity, we may replace $\frac{r}{r+s}$ with γ; therefore, $\frac{s}{r+s}$ becomes $1 - \gamma$.We show the space of trust vectors in Fig. 2.

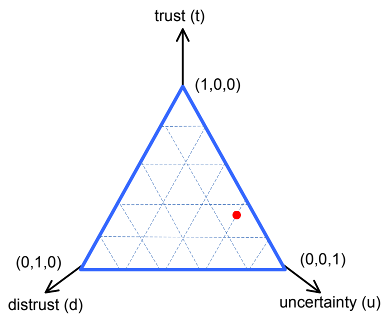


Figure 2. The space of trust vectors, $\bar{V}$, forms a *simplex* (triangle), where every point on the simplex satisfies the property that $t + d + u = 1$. The dot indicates the position of an example vector, $\langle 0.2625, 0.0875, 0.65 \rangle$, in this simplex.

3 Predictors of Positive Experience

Our trust model is based on the confidence in the probability of having a positive experience with a WS i, which is updated using the number of positive and other types of experiences with the WS. Unlike previous efforts [9, 16], we hypothesize what is meant by an 'experience' with i. Although experiences tend to be subjective, the variables we provide are meant to serve as *predictors* of likely positive experiences. These predictors are not exhaustive and others may also exist.

While interacting with a WS, a user may consider the competency, reliability and honesty of the service. We define an experience as:

Definition 4 (Experience)**.** *Experience of a user with a WS, i, is the behavior of i in terms of its competency, reliability and honesty.*

• *Competency:* ability of the service to perform all the necessary actions to achieve a particular goal.
• *Reliability:* ability of the service to provide the same desired output upon repeated invocations without any discrepancy.
• *Honesty:* ability of the service to be faithful to terms agreed upon in the service level agreement and other contracts.

We may expect experiences of users with a WS to be positive if the WS is competent, reliable and honest. Therefore, we briefly discuss how we may objectively ascertain these characteristics of WSs.

We may measure honesty of a service as the difference between the advertised or agreed upon values of QoS parameters appearing in the service level agreements and the actual observed values of the QoS parameters. As cost is not observed, we limit our focus to the parameters, response time $\bar{R}$ and availability A. The response times are normalized, as we show below, to make them comparable with availability. Let A_a and $\bar{R}_a$ be the advertised values while A_m and $\bar{R}_m$ be the observed values of the QoS parameters. Then, we define an objective measure of honesty, $h \in [0, 1]$, as:

$$h = 1 - \frac{|A_m - A_a| + |\bar{R}_m - \bar{R}_a|}{2}$$

where $\bar{R} = \frac{R - R_{min}}{R_{max} - R_{min}}$ if $R_{max} - R_{min} \neq 0$, otherwise 1; and R_{max}, R_{min} are the maximum and minimum values of response times respectively, among all available services. The differences could be assumed to default to zero if the observed QoS values improve on the advertised ones. Higher values of h indicate an honest service.

Reliability, r, is measured as one minus the fraction of times the service fails or does not behave as per its function measured over as long a sequence of invocations as possible. Finally, we formalize competency, c, as a binary valued concept indicating whether the service is able to satisfy the goals. If it does, we assign $c = 1$, otherwise 0.

4 Derived Trust for Compositions

Our aim is to allow existing composition tools to be utilized in conjunction with our approach. WSC algorithms often generate multiple compositions that satisfy the requirements. A trust vector for each of these compositions is derived, and the composition with the highest trust ratio – deemed most trustworthy – is selected for execution. Consequently, the selected composition not only satisfies the functional and QoS requirements, it is also the most trustworthy among the candidates, and most likely to perform as expected.

In order to derive the composite trust, we consider *four* types of basic flows of services that are often encountered in compositions. For simplicity, we consider a composition of two services, w_1 and w_2, in the different configurations, with beliefs over the probabilities of having positive experiences with them, $B(p_1; a_1, b_1)$ and $B(p_2; a_2, b_2)$, respectively. The methods may be generalized to more services in a straightforward way. Our approach is to compute the belief over the probability of having a positive experience with the WSC. Given the belief density and a way of deriving the positive experiences for the WSC, we may compute the certainty level and, subsequently, the trust vector as mentioned in Section 2.

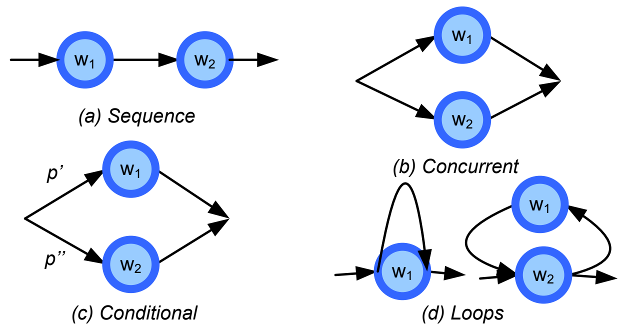


Figure 3. Different flow constructs typically appearing in a composition. We derive a trust vector for each construct.

• *Sequential flow:* Each of the services in a sequential flow (see Fig. $3(a)$) is executed. Hence, a positive experience with the WSC is contingent on positive experiences with each of the two component WSs. If executions of the WSs are independent of each other, then the probability of a positive experience with the WSC, p_c, is the product of the individual WS probabilities: $p_c = \Pi_{i=1}^2 \, p_i$

Note that because p_c is a product of the individual probabilities, we could view the range of p_c as a product space. Given the individual beliefs over the probabilities p_1 and p_2, we may derive the belief density over p_c:

$$\begin{aligned} B^s(p_c) &= \int_0^1 B(p_1; a_1, b_1) B(\tfrac{p_c}{p_1}; a_2, b_2) \tfrac{1}{|p_1|} \, dp_1 \\ &= B(p_1; a_1, b_1) \otimes B(p_2; a_2, b_2) \end{aligned} \quad (3)$$

We introduce the operator $\otimes$ for simplicity of presentation. The above equation is an application of a well-known way of obtaining the probability density function over a product space of random variables (see Theorem 7, pg. 141 of [14]). Although the components are beta densities, $B^s(p_c)$ may no longer be a beta density.

• *Concurrent flow:* We show a concurrent flow in Fig. $3(b)$. Analogous to the sequential flow, each of the services in a parallel flow must also be executed. Hence, p_c is derived analogously to the one for the sequential flow. Consequently, the belief density is obtained as shown previously in Eq. 3.

• *Conditional flow:* As shown in Fig. $3(c)$, any one of the branches is executed in a conditional flow. For example, let there be two branches that are followed with probabilities, p' and p'' ($p' + p'' = 1$). These probabilities may be obtained from prior interactions. Then, the probability of a positive experience with the WSC is a weighted sum of the probabilities for the individual WSs: $p_c = p' \times p_1 + p'' \times p_2$

Given the individual beliefs over the probabilities p_1 and p_2, we may derive the belief density over p_c as:

$$B^o(p_c) = \frac{1}{p'p''} \int_0^1 B(\frac{p_1}{p'}; a_1, b_1) B(\frac{p_c - p_1}{p''}; a_2, b_2) \, dp_1 \quad (4)$$

We again refer to Theorem 7 on pg. 141 of [14] for proof of the above equation when p_c is composed as shown previously.

• *Loop:* We restrict our analysis to loops that are iterated a fixed number of times, say n. Analogous to the sequential flow, each WS in a loop is executed n times (Fig. $3(d)$). Therefore, for the case where two services participate in a loop, let, $p'_c = p_1 \times p_2$. Then,

$$B^s(p'_c) = B(p_1; a_1, b_1) \otimes B(p_2; a_2, b_2)$$

where the operator $\otimes$ is as defined in Eq. 3. For the WSC: $p_c = \Pi_1^n p'_c$, thus:

$$B^{l(n)}(p_c) = \big((B^s(p'_c) \otimes B^s(p'_c)) \otimes B^s(p'_c)\big) \otimes \ldots n \text{ times} \quad (5)$$

This may be extended to more services within the loop easily.

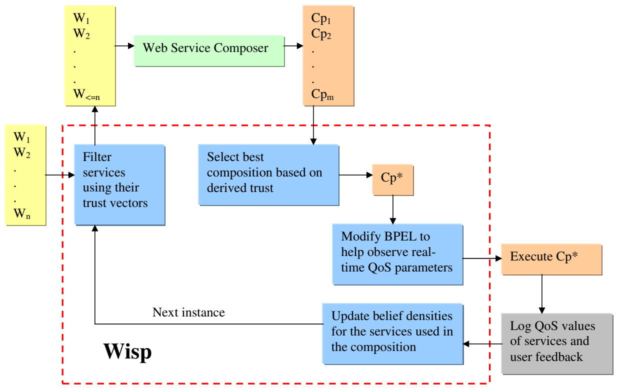
As a composition may consist of one or more of these basic flow constructs and we have mathematically shown a way to derive the belief density for each of these constructs, the aggregate belief over the probability of a positive experience with the entire composition can be computed. We point out that these methods for deriving the composite probability for a WSC, p_c, are loosely analogous to calculating the aggregate QoS parameters of a composition as shown in [4].

An algorithm for computing the distribution over the product (ie. implementing the $\otimes$ operator) is mentioned in [7]. However, exact computation of the distribution often turns out to be complex; approximations provide reasonably close distributions in significantly less time. We utilize a sampling scheme to generate the density over the product space, which converges to the exact as the number of samples approaches infinity. We present the sampling algorithm in Fig. 4. We first sample the individual densities and tabulate the frequencies of the product of the two independent variables. The frequency histogram is converted into a normalized cumulative distribution function (cdf), which is used to obtain an approximate probability density function.

As we mentioned before, we may compute the certainty of the belief density as shown in Section 2.3. What remains is how much of the certainty should be allocated to trust and distrust. This is equal to the proportion of positive, γ_c, and non-positive experiences, $1 - \gamma_c$, with the composition, respectively. Because users interact with the individual WSs in the composition, we must *estimate* the proportions from the experiences with the individual component WSs. As with the computation of p_c, the estimation is contingent on the type of flow in the WSC. For flows where all component WSs are executed (sequential, parallel and loop in Fig. 3), a positive experience with each of the component WSs is very likely to translate into a positive experience with the WSC. Hence,

```
Algorithm for $B(p_1; a_1, b_1) \otimes B(p_2; a_2, b_2)$
  $n$, $numBins$   //number of samples and bins
  $frequencyCountBin[1..numBins]$ //freq. count for hist.
  $cdf[1..numBins]$, $pdf[1..numBins]$
  Sample densities and tabulate freq. of product of samples
  for $i$ = 1 to $n$
      Sample $s_1 \sim B(p_1; a_1, b_1)$, $s_2 \sim B(p_2; a_2, b_2)$
      $p \leftarrow s_1 \times s_2$
      Increment $frequencyCountBin[p]$ by 1
  Convert to a cdf and then to a pdf
  for $i$ from 2 to numBins
      $cdf[i] \leftarrow cdf[i-1] + frequencyCountBin[i]$
  for $i$ from 2 to $numBins$
      Use the inverse of trapezoidal rule for numerical
      quadrature with values in $cdf[i]$, $cdf[i-1]$ to find $pdf[i]$
```

Figure 4. Sampling algorithm for approximating a probability density over a product of 2 independent random variables.



W_1
W_2
.
.
.
$W_{<=n}$
Web Service Composer
Cp_1
Cp_2
.
.
.
Cp_m
W_1
W_2
.
.
.
W_n
Filter services using their trust vectors
Select best composition based on derived trust
Cp*
Modify BPEL to help observe real-time QoS parameters
Execute Cp*
Next instance
Update belief densities for the services used in the composition
Log QoS values of services and user feedback
Wisp

Figure 5. Details of Wisp – our framework for deploying trusted WSCs. Note that Wisp and the WS composition tool are decoupled and may be located on different hosts.

the proportion of positive experiences with the WSC is at most the *minimum* of the proportions of positive experiences among all component WSs. We let γ_c be the minimum proportion. Next, we consider the case where the flow involves a condition (see Fig. $3(c)$). If γ_1 and γ_2 are the proportion of positive experiences for WSs w_1 and w_2, respectively, then $\gamma_c = p'\gamma_1 + p''\gamma_2$. Given γ_c, we may obtain the trust vector of the composition as shown in Definition 3.

5 Wisp

We present a new framework called Wisp for integrating trust considerations into WSCs and selecting trusted WSCs for deployment. While traditional composition techniques form compositions that meet the functional and non-functional requirements in theory, the compositions may not behave accordingly in practice. Wisp seeks to reduce this pragmatic gap by associating trust vectors to WSCs and updating the trust vectors based on user feedback. We describe the specifics of Wisp below and show its design in Fig. 5.

5.1 Filtering Untrustworthy Services

For each WS, i, available for composition, Wisp associates and maintains the belief density over the probability of having a positive experience with i. Because there is complete uncertainty about i initially, the belief density is initialized to a flat line. Wisp measures the certainty level of the belief density and computes the trust vector for i indicating the trust (t_i), distrust (d_i) and any remaining uncertainty (u_i) about having a positive experience with i, as mentioned in Section 2.

Wisp allows a preliminary filtering step, in which a target user of the WSC may assume a trust ratio threshold, $tt \in [0, 1]$. Services available for composition whose trust ratios fall below the threshold, $\frac{t_i}{t_i+d_i} < tt$, are deemed untrustworthy by the user, and immediately filtered out. The trust threshold may vary with WSs – critical WSs may warrant higher thresholds than non-critical ones – and it need not be fixed but may vary over time per the judgment of the user.

We point out two advantages in allowing a trust threshold. First, it allows the user to decide a minimum quality level below which services cannot participate in the composition based on prior interactions. Second, by reducing the set of candidate services, the composition may occur in lesser time due to a reduced search space.

5.2 Deploying Trusted Compositions

Any WSC tool could be utilized to generate compositions of the services that theoretically meet requirements. Often, multiple satisfying compositions are produced. Wisp seeks to deploy one of these compositions for execution that is expected to perform well in practice. For each of these compositions, Wisp derives the aggregated belief density over the probability of having a positive experience with the composition. As we mentioned in Section 4, the derivation is based on the type of flow present in the WSC. Given the belief density, we may compute the certainty and subsequently, the trust vector for each of the compositions.

Let the WSC tool produce m compositions and each is associated with a trust vector, $\bar{v}_j,\ j = 1 \ldots m$. Wisp chooses a WSC, Cp^*, to execute using the following algorithm:

1. Among the m compositions, select Cp^* to be the WSC j which has the highest proportion of certainty allocated to trust in its trust vector, $\bar{v}_j$:

$$Cp^* = \arg\max_{j=1\ldots m} \frac{t_j}{t_j + d_j}$$

2. If there exists a tie, select the WSC j among the tied ones to be Cp^* which has the highest level of certainty, $t_j + d_j$, in its trust vector.
3. Compositions that remain tied have identical trust vectors. Finally, break ties randomly to obtain Cp^*.

In order to uncover possible correlations between user feedback and observed QoS parameters of compositions, Wisp supports the logging of response times and availability data of component WSs in Cp^*. Specifically, Wisp parses the implementation log to obtain the response times. Availability of a service is ascertained by noting whether a valid response is received over repeated invocations.

5.3 Updating Belief

The composition, Cp^*, chosen as mentioned in the previous subsection is deployed. Wisp provides an intuitive interface to users for collecting feedback on the individual services participating in the composition. In particular, users may indicate whether their experiences with the component services in the deployed WSC have been positive. These positive and non-positive feedbacks for each service in the WSC, say i, form the tuple, $\langle r, s \rangle$, which is used to update the belief density over p_i, as in Section 2.2. Belief densities for all WSs in the deployed Cp^* are updated using the feedbacks. Wisp indexes the computed trust vectors of the WSs in a hash table for quick lookup during the next instance of its usage.

Note that trust vectors associated with a service are not private to a user and are updated by all users interacting with compositions containing the service. This has the advantage that a service deemed untrustworthy by some user ($\frac{t_i}{t_i+d_i}$ is below her threshold, tt) could have its trust ratio improve over time because of interactions with other users, until it meets the threshold of the user. In other words, the update of the trust in a service allows the service to possibly migrate from the filtered list to the unfiltered one, if its performance improves.

6 Experiments

Pilot Study We conducted a small study to explore possible correlations between users' assessment of a positive or non-positive experience with WSs in a composition and our hypothesized variables – honesty, reliability and competency of the WSs – as mentioned in Section 3. The subjects were volunteer graduate students in the department of computer science at the University of Georgia whose prior experience interacting with WSs ranged from significant to none. *Fifteen* subjects responded to our call and participated in the study.

Each subject was presented with 10 randomly selected sequential compositions of 3 WSs each. To maintain realism, the compositions were deployed using the ActiveBPEL engine[1], the WSDL WSs using the Apache server and inputs were provided to the services using soapUI [2]. The subject was informed about the advertised response times, which ranged from 1,000ms to 1,500ms, and availability rates, ranging from 0.75 to 1.0, of the WSs participating in the compositions. Subjects witnessed 4 executions of each of the compositions and were shown the observed response times and availability rates of the WSs averaged over the 4 runs. Some of the WSs in the compositions were programmed to deviate from their advertised parameters by varying levels. Subjects were then asked to rate their opinion of the WSs in the composition as 'positive' or 'non-positive' in a computerized questionnaire.

Given the advertised and observed values of the QoS parameters, the honesty and reliability of each WS in the 10 compositions was calculated (see Section 3). Note that the subjects were not made aware of the variables and their computations. For the sake of simplicity, we assumed that each WS is competent. These variables were then combined to produce a single predictor variable as follows:

$$\text{predictor variable} = \frac{h + r + c}{3}$$

Note that the predictor variable ranges from 0 to 1, and higher values of the variable signify WSs that are honest, reliable and competent. In Fig. 6 (a)–(d), we show the preference models of 4 of the subjects who participated in the study. The '-' and '+' symbols indicate the subject's non-positive or positive opinion about a WS and their positions on the horizontal axis indicate the value of the predictor variable for that WS. Each model contains 30 such data points, many of them too close to distinguish them clearly.

We utilized a decision tree classifier, J48, available in the Weka machine learning package[3], to find a threshold value of the predictor variable that best separates positive from non-positive experiences. For most subjects, finding the partition was possible as their feedbacks were consistent. However, see Fig. 6(a) for an exception where some WSs predicted to offer a good experience were disliked by the subject. The threshold in each model is indicated by the vertical dashed line.

We draw several preliminary *conclusions* from this pilot study: (i) Because WSs having high predictor variable values were consistently rated by subjects to offer a 'positive' experience, a statistical correlation between our hypothesized predictors and actual experiences exists. (ii) Further exploration of the exceptions in Fig. 6(a) revealed that the WSs rated 'non-positive' met their advertised response times but fell slightly short of the advertised availability rates. This suggests that users could be attaching unequal importance to the different QoS parameters – a conclusion that requires further study. (iii) Finally, as we may expect, users exhibited differing bars below which they judged their experience with WSs to be not positive. However, these thresholds were strictly higher than 0.5. Interestingly, subjects that did not have prior experience with WSs displayed lower thresholds.

Evaluation of Wisp Our objective is to validate the utility of our trust framework. We empirically demonstrate that a consideration of trust in the selection of compositions results in compositions that progressively exhibit less deviations from their advertised or agreed upon QoS values.

[1]ActiveBPEL: http://www.activevos.com/community-open-source.php
[2]soapUI: http://www.soapui.org
[3]Weka: http://www.cs.waikato.ac.nz/ml/weka/

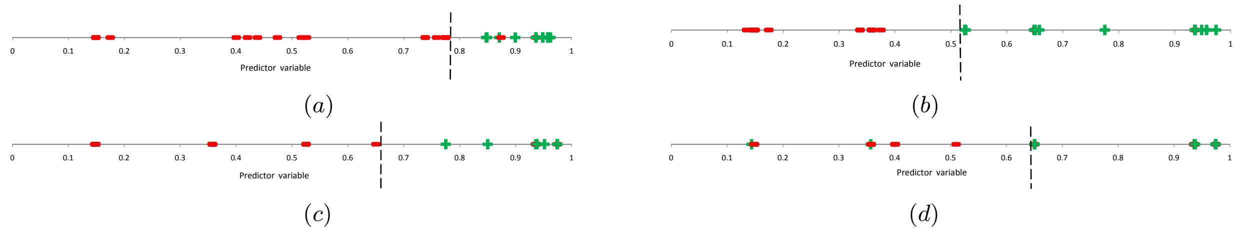


Figure 6. (a) – (d) Preference models of 4 subjects out of 15 that participated in the pilot study. '-' (in red) and '+' (in green) symbols indicate non-positive or positive assessments of their experiences with WSs participating in the compositions. The vertical line reflects the best partition of the hypothesized predictor variable.

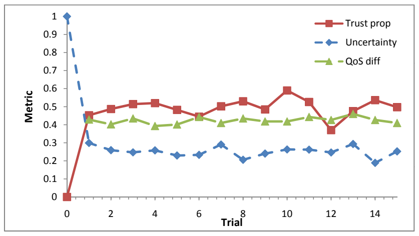


Figure 7. Trust proportion, uncertainty and composite QoS difference across trials.

We used 5-step sequential, concurrent and looping compositions with a choice of 2–3 WSs identical to those used in the study, at each step. Thirty-two distinct compositions of the services were utilized. WSs are described using WSDL and the compositions using WS-BPEL. We utilize ActiveBPEL to deploy and soapUI to execute our compositions. Wisp is deployed as a WS and it retrieves details of execution of services from the process log of the Web server. For our experimental environment, one-third of the WSs were programmed to significantly deviate from their advertised QoS parameters. We did this by adding redundant CPU cycles in the services to slow down the response times and a special message to indicate unavailability. As compositions were executed, positive or non-positive feedback was assigned to services by following the user preference models that were elicited previously. In this way, we simulated actual users of the compositions.

After executing a composition, the updated trust vector was computed as mentioned in Section 5.3. Once the belief densities of the services are updated, the composite trust vectors of each of the 32 compositions were computed as shown in Section 4. We selected the composition to deploy using the algorithm mentioned in Section 5.2.

In Fig. 7, we show the trust proportions, uncertainties and difference between the composite empirically measured QoS of the deployed compositions and the composite advertised QoS of the composition. The composite QoS values were computed as per the rules in [4]. We executed 15 trials, averaged over 5 iterations, during which each composition was executed 4 times. The empirically observed QoS parameters were averaged over these 4 runs. Because of the widely differing user preference models, we do not observe a significant increase in trust proportion or a decrease in QoS difference for the selected compositions across the trials.

7 Related Work

A vast literature exists to explain trust from different perspectives. To remain within the scope of this paper, we provide previous characterizations of trust followed by a few trust models, trust within the scope of WSs and lastly trust in WSC.

Social definitions of trust Gambetta [6] defines trust as a particular level of the subjective probability with which an agent performs a particular action, before it can monitor the action and in a context in which it affects others' actions. McKnight and Chervany [12] characterize a trusting intention as the extent to which one party is willing to depend on the other party in a given situation with a feeling of relative security, even though negative consequences are possible. Olmedilla et al. [13] define trust in the field of WSs as the measurable belief that provider behaves dependably for a specified period within a specified context in relation to a WS. Our characterization is loosely based on all the three definitions. While the above definitions are vague about the characteristics of the trusted entity, which makes the trusting entity trust it, we ground experiences with WSs along three axes: honesty, reliability and competency. Trust is also often classified into direct trust, which a user has on another user or service, and recommender trust [1, 3], which a user has on another user or service that recommends others. Our focus in this paper is on modeling direct trust, although pursuing recommender trust is one avenue of our future work.

Trust in WS computing In the area of WSs, the WS Trust Language aims to enable applications to construct trusted message exchanges. We do not focus on authentication mechanisms and assume that the concerned participants are legitimate providers or requesters. Liu et al. [10] introduced a dynamic QoS computation model by means of a central QoS registry. Their broker architecture is human-oriented – consumer WSs give feedback to provider WSs. Solely relying on user feedback has a two-fold disadvantage: different users may have different opinions and a service preferable to one consumer might not appear so for another. Also, some users might give a deceptively negative or positive feedback on purpose. Thus, a way to correlate the subjective feedback of users with objective measures is beneficial. Adam et al. [2] introduce a trust index for a service provider and requester, which is dynamic and propagated throughout the environment. They provide means of determining if a service provider or requestor violates the WS usage policies specified. In comparison to [2], we emphasize the fact that we associate trust with a particular service and not with the service provider. This is because users interact with a service and providers may have both low and high quality services.

Trust in WSC Singh [15] discusses the formation of a trust network among agents who provide services to each other. Agents help each other locate trustworthy services to include in a service composition. While Singh focuses on settings leading to service compositions, our work is more focused on identifying trustworthy compositions. Furthermore, Singh does not provide a concrete model of trust beyond a discussion of its desirable properties. Zu et al. [17] propose a framework for reputation-enhanced QoS-based WS discovery that extends the UDDI registry to publish the QoS information of WSs and uses a reputation manager to assign reputation scores to the WSs based on customer feedback. A significant difference from our approach is the focus of [17] on discovering reputable WSs, while Wisp focuses on building trusted WSCs. Another distinction is their approach of storing all previous reputation scores, which may lead to considerable storage overhead over time. However, it has the advantage of allowing flexibility in length of the history of reputation scores that we could consider. In contrast, we update the trust using a Bayesian approach, which forms a sufficient statistic for the entire history of user feedbacks.

8 Discussion

We conceptualized trust in WSs using a mathematical model that meets much of our intuition: (i) It permits modeling our uncertainty about trusting or distrusting WSs initially when no prior experience with them exist. (ii) As the number of experiences with a WS increases though the proportion of positive experiences, γ, may remain fixed, the certainty level increases. Our certainty is higher if we have 42 positive experiences out of 50 in comparison to 8 positive ones out of 10. (iii) Furthermore, the certainty is lowest when an equal number of positive and non-positive experiences obtain. A useful refinement is to integrate the reputation of the users into the update [11]. This could help in diminishing the impact of feedbacks given by malicious users.

References

[1] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *HICSS*, pages 4–7, 2000.

[2] N. R. Adam, A. Kozanoglu, A. V. Paliwal, and M. Youssef. Mutual trust in open environment for cascaded web services. In *ACM Workshop on SWS*, pages 107–108, 2006.

[3] T. Beth, M. Borcherding, and B. Klein. Valuation of trust in open networks. In *ESORICS*, pages 3–18, 1994.

[4] J. Cardoso, A. Sheth, J. Miller, J. Arnold, and K. Kochut. Quality of service for workflows and WS processes. *JWS*, 1, 2004.

[5] S. Dustdar and W. Schreiner. A Survey of Web Services Composition. *Intl. J. of Web and Grid Services*, 1(1):1–30, 2005.

[6] D. Gambetta. Can we trust trust? In D. Gambetta, editor, *Trust: Making and Breaking Cooperative Relations*, pages 213–237. Basil Blackwell, 1988.

[7] A. Glen, L. Leemis, and J. Drew. Computing the distribution of the product of two continuous random variables. *Computational Statistics and Data Analysis*, 44(3):451–464, 2004.

[8] E. T. Jaynes. Where do we stand on maximum entropy. In Levin and Tribus, editors, *The Maximum Entropy Formalism*, pages 15–118. MIT Press, 1979.

[9] A. Josang. A subjective metric of authentication. In *ESORICS*, pages 329–344, 1998.

[10] Y. Liu, A. H. Ngu, and L. Zeng. Qos computation and policing in dynamic WS selection. In *WWW*, pages 66–73, 2004.

[11] Z. Malik and A. Bouguettaya. Evaluating rater credibility for reputation assessment of WSs. In *WISE*, pages 38–49, 2007.

[12] D. H. McKnight and N. L. Chervany. The meanings of trust. Technical report, MIRC, University of Minnesota, 1996.

[13] D. Olmedilla, O. F. Rana, B. Matthews, and W. Nejdl. Security and trust issues in semantic grids. In *Dagstuhl Seminar on Semantic Grid*, no. 05271, 2005.

[14] V. K. Rohatgi. *Introduction to Probability Theory and Mathematical Statistics*. Wiley Interscience, 1976.

[15] M. Singh. Trustworthy service composition: Challenges and research questions. In *Workshop on Trust, Reputation and Security: Theory and Practice, AAMAS*, 2002.

[16] Y. Wang and M. P. Singh. Formal trust model for multiagent systems. In *IJCAI*, pages 1551–1556, 2007.

[17] Z. Xu, P. Martin, W. Powley, and F. Zulkernine. Reputation-enhanced qos-based web services discovery. In *ICWS*, pages 249–256, 2007.