Agent-Based Trust Model Involving Multiple Qualities

E. Michael Maximilien
IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120
maxim@us.ibm.com

Munindar P. Singh
North Carolina State University
Department of Computer Science
Raleigh, NC 27695
singh@ncsu.edu

ABSTRACT

A key limitation of current Web services standards is the inability to differentiate service instances at runtime using (nonfunctional) qualities of services (QoS). Such differentiation is necessary to allow for runtime selection and binding to service instances in a manner that continually adapts selected services to the service consumer's preferences and needs. Quality values are volatile, depend on the trust associated with instances and service providers, and also depend on the needs of service consumers. We propose a multiagent framework where agents consider the consumers' QoS preferences, determine trust levels to associate to service instances and providers, and automatically select service instances on a consumer's behalf. The service agents use a trust model that is centered on a shared conceptualization for QoS (ontology) and a QoS preference model that considers consumer's tradeoffs among qualities as well as relationships between qualities. We evaluate our approach via simulations on simplified but realistic service instances and service consumers. Our results show that using these considerations for QoS, service agents are able to determine over time the 'best' service selection for a consumer.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multiagent systems*; D.1.0 [**Software Engineering**]: Programming Techniques—*general*

General Terms

Algorithms, Reliability, Experimentation

Keywords

Service-Oriented Architecture, Service-Oriented Computing, Semantic Web Services, Service selection, Multi agent systems, Quality of Service, Trust

1. INTRODUCTION

Web services standards, while essential for automated access to services on the Web, are mostly geared towards design-time automation and usage. Limiting ourselves to design-time automation is not appropriate for software agents doing automatic runtime service selection and binding. Service descriptions via WSDL are necessary but not sufficient to automate service selection and binding—they principally lack a means to represent nonfunctional service attributes. To achieve true runtime integration we need a means to collect nonfunctional information about services and use that information to assign dynamic trust levels to the service providers and implementations.

As in real-life situations, the agents in our system need to be able to make decisions using partial information, limited computational resources, and bounded time. As such, an agent's decision process is to derive a *trust* value for each service instance. The service instance with the highest trust value is selected.

Our primary contribution is a trust model based on a shared conceptualization of qualities of service (QoS). Our trust model takes into account a consumer's subtle preferences for qualities when determining the trust value to assign to service instances. Specifically, we consider a consumer's preference for tradeoffs among qualities and relationship between qualities. We show the effectiveness of our approach using simulation experiments.

The remainder of this paper is organized as follows. Section 2 reviews the conceptual background underlying our approach, including assumptions and framework. Section 3 formulates the service selection problem using trust. Sections 4 and 5 give two solutions to this problem. Section 6 discusses our empirical evaluation and shows our results. Section 7 highlights some related work. Section 8 discuses some directions.

2. BACKGROUND

The trust assigned to a service instance is, in essence, a predictor of quality values for future usage of the service instance. It is an aggregate value that corresponds to historical levels of quality of the service instance and to how well the provider's quality advertisements match the quality needs of a consumer. Our approach to selection extends the framework and trust model described previously [6, 7].

Since we are mainly concerned with agents that automate service selection decisions on behalf of their consumers, we make certain simplifying assumptions about the environment and the agents participating in it.

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.

Assumption 1 (Service Agent Environment). *The agent environment is open and dynamic, with no central authority.*

Although our agents are self-interested and autonomous they gain value from sharing and aggregating their opinions. Sharing allows them to gain a global view of the quality and trust of the service instances and service providers.

Assumption 2 (Shared Quality Opinion). *The agents are free to share opinions on service interactions and this information is accessible to all agents.*

We also further assume that trust values are scaled appropriately in their calculation to allow them to be added to determine a composed value. Agents contribute their subjective and objective quality opinions to common agencies. Like other practical works on reputation, this approach assumes that the opinions are truthful.

Assumption 3 (Truthful Agents). *Agents report true assessments of nonfunctional attributes that they gathered by interacting with services.*

This assumption is realistic for two principal reasons. First, the consumer service agents gain from the collected aggregated quality values, so individually they do not have a direct incentive to be untruthful. Finally, since in practice the population of consumer agents is likely large and few agents will be untruthful the aggregated values are not significantly impacted by untruthful reporting. A possible risk is that some providers could have consumer agents that generate false ratings on their behalf, and at a volume that overwhelms honest evaluators. We consider this case out of the scope for our current formulation and an area for future research.

We now introduce our main trust concepts. Sections 4 and 5 build on these definitions to create our trust model.

Provider and Service Trust. A value for predicting the qualities of service—those that matter to a service consumer—for a potential interacting service provider and its service implementations.

Quality Reputation. An aggregate value for a quality of a service implementation over time and over many service consumer interaction episodes.

We distinguish three phases of interactions between consumers and services: discovery, selection, and binding. Figure 1 shows a service instance usage lifecycle, from discovery to binding, and the typical steps in between resulting from the different types of interactions between consumers and providers. Once a service interface is discovered, a consumer selects an implementation from one of multiple implementations from different providers. Whereas service discovery requires matching a service functional attributes, service selection involves QoS. Once an implementation is selected it is bound to for usage. During usage, a consumer may have a policy to rebind or reselect the service implementation, e.g., if the current implementation's QoS falls below some threshold.

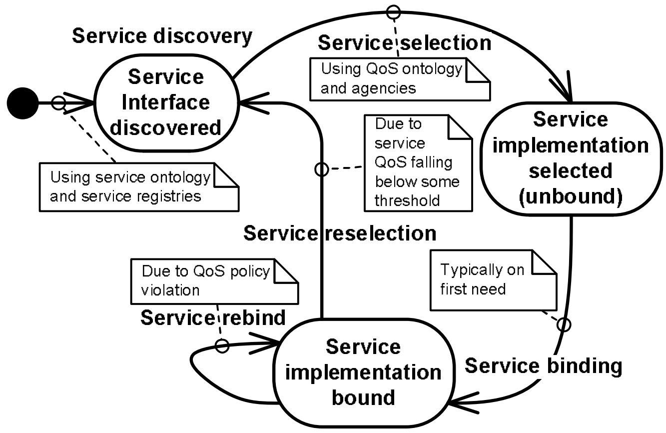

Figure 1: Service instance usage life-cycle.

3. PROBLEM FORMULATION

To formulate the service selection problem, we begin with definitions of the basic objects and then give a generic definition of the selection problem as a trust optimization problem (introduced below). Using this general problem description, Sections 4 and 5 incrementally discuss our service trust model, thereby providing an abstract solution to the selection problem.

Let Υ stand for the set of all URIs and let $\Upsilon_S \subset \Upsilon$ be the set of URIs representing services on the Web. Let Δ be the set of all application domains. An example of an application domain is *Math*—representing services for mathematical calculations. A Web service is a triple $s = (\iota, d, i)$. Where $\iota \in \Upsilon_S$ is the interface, $d \in \Delta$ is the application domain, and $i \in I_s$ is the implementation, where $I_s \subset \Upsilon_S$ is the set of all implementations of ι.

We associate nonfunctional attributes, in the form of qualities, with services. For example, *Capacity* is a service quality representing the maximum number of concurrent consumers supported by a given service and *ResponseTime* is the request-to-response time for a given service. A quality ontology [6] gives a comprehensive overview of the service qualities that we are considering.

Let Q be a service quality or a nonfunctional attribute. For a particular implementation i, $\overline{Q}$ is the set of values for that particular quality, contributed by the agents that selected and used the implementation. Let Φ be the set of all qualities and $\Phi_d \subseteq \Phi$ be the set of qualities applicable to the application domain d.

We also associate *directionality* with each quality Q. In notation, $dir(Q) = \uparrow$ indicates that the quality Q is directionally increasing which means that higher values for Q are preferred. And $dir(Q) = \downarrow$ indicates that lower values are preferred.

The service selection problem reduces to picking the most trusted service. This is reasonable because the trust assignment depends on the service implementation's qualities. Specifically, our service agents need to dynamically associate a trust value with each service implementation and select the service implementation with the highest assigned level of trust. We associate trust with an implementation using its qualities, while taking into account the quality preferences of the consumer, and the quality advertisements of its provider.

A consumer c is a pair (ι, Pref), where ι is a service interface and Pref are the consumer's quality preferences for the implementations of ι. A service provider p is a triple $(\iota, \vec{i}, \vec{Adv})$, (with n as the number of implementations of interface ι by p) where $\vec{i} = \langle i_1, \ldots, i_n \rangle$ is an ordered set of the implementations of ι, and $\vec{Adv} = \langle Adv_1, \ldots, Adv_n \rangle$ is an ordered set of advertisements—one for each implementation.

DEFINITION 1 (SERVICE SELECTION). *Let P be the set of all providers with implementations for service s. Our problem is to select the service implementation $i \in I_s$ of service s from all service providers $p \in P$ such that:*

$$i = \arg\max_{i \in I_s} \{\mathit{trust}(i, \Phi_d)\}$$

Where $\mathit{trust}() : I_s \times \Phi \mapsto \mathcal{R}$ is a service trust function.

The *trust* function takes into account the service implementations, the service's domain qualities, the quality preferences of consumers, and the quality advertisements of providers. We provide abstract solutions to this problem in the following sections.

4. SERVICE TRUST MODEL: SIMPLE PREFERENCES

We begin with an abstract solution to the service trust function that considers simple preferences of consumers. Section 5 expands on this initial solution to consider a consumer's more complex quality preferences.

In order for an agent to make sense of the collected data we need to aggregate it into one value—the quality reputation for the implementation. Since the quality exhibited by an implementation varies with time, and recent opinions are a better indication of current qualities, we need to make sure that recent opinions count more than older ones. Similar to Zacharia and Maes [13], we achieve the time discounting of quality values by dampening the collected values.

DEFINITION 2 (SERVICE QUALITY REPUTATION). *The reputation R_{Qi} of an implementation, with respect to quality Q, is the aggregation of the quality values for the implementation i of service s over some time interval. Specifically,*

$$R_{Qi} = \frac{1}{n} \sum_{k=0}^{n} q_k \delta^{-t(q_k)}$$

where $\overline{Q} = \{q_k\}_{k=0}^{n}$ is the set of quality values collected from agents that selected implementation i, $\delta \in R$ is the quality Q's dampening factor, and $t() : \overline{Q} \mapsto \mathcal{Z}^+$ is the time for which the quality value q was collected. $t(q) = 1$ for the most recent collected value and $t(q) > 1$ for all other values.

Based on a service implementation's quality reputation R_{Qi} as well as the consumer's quality preferences Pref, in order to have a solution to the selection problem of Definition 1, we need to complete the formula for $\mathit{trust}()$. Specifically, we need to derive a *trust* function using R_Q while taking into account the consumer's preferences $\vec{\phi}$, and the provider's advertisements $\vec{Adv}$.

Since we can represent a consumer's trustworthiness assignment to a provider, to complete the implementation of the *trust* function of Definition 1, we need a way to assign a trust value to a particular service implementation taking into account the consumer's preferences.

We achieve a trust assignment to an implementation by matching the consumer's quality preferences to the implementation provider advertisement and assigning a degree to the match. The matching degree takes into account the quality advertisements, preferences, and reputation [7].

5. SERVICE TRUST MODEL: COMPLEX PREFERENCES

The previous section presented a solution to the selection problem by introducing a *trust* function that takes into account the quality reputations of implementations and preferences of consumers. Though reasonable, this solution is lacking in the way it represents the quality preferences of consumers. Specifically, it does not account for relationships between qualities and a consumer's preferences for tradeoffs among qualities.

First, the qualities exhibited by an implementation typically represent the tradeoffs made by providers when exposing their services; as a consequence, qualities will sometimes have relationships with each other. Second, since the qualities exposed by an implementation are the results of tradeoffs made by the provider—a provider cannot realistically maximize all desirable qualities for an implementation—it follows that to truly capture a consumer's quality preferences, we also need to understand the consumer's relative preferences for one quality as well as among qualities.

To address these issues, we present a preference model for service consumers that takes into account quality relationships and the consumer's quality tradeoffs.

5.1 Consumer Quality Utility

To derive a better representation of the consumer's utility over all qualities, we need to understand the consumer's utility for any of its preferred qualities, the consumer's preferences (if any) over tradeoffs among qualities, and the relationships between qualities.

The consumer preferences vector $\vec{\phi}$ currently accounts for the consumer's preferences for qualities based on boundary conditions and an ideal preferred value for each quality. To complete the representation, we additionally need to have an understanding of the consumer's utility for each quality and the shape of the utility function. For instance, in addition to the preferred value for a quality, is there a limit for the quality values for which the consumer becomes indifferent to further improvements in the quality? That is, assume that a consumer's application is limited to a *Capacity* of 50 number of concurrent user connections and makes use of a *LoanService* service. A *LoanService* implementation s_1 whose advertised and reputed *Capacity* quality is 500, is probably of no benefit to the consumer. However, another service implementation s_2 whose *Capacity* is 51, can result in a better match; especially if s_2's other qualities also match the consumer's remaining quality preferences. We say in this case that the consumer's utility for the *Capacity* quality exhibits a saturation-limit utility shape—i.e., the consumer's utility saturates after a limit value for the quality.

5.1.1 *Shape of Consumer's Quality Utility*

We consider two shapes for a consumer's utility for a qual-

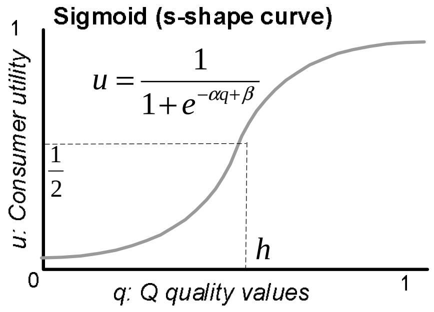

Figure 2: Sigmoid consumer quality utility.

ity:

1. *Linear.* Where the consumer's utility for a quality is unbounded.

2. *Sigmoid.* With this shape, we can consider qualities for which the consumer's utility is nil up to a certain threshold value, and then quickly increases and flattens out again. This shape is particularly important to represent a consumer's utility for qualities where the values must be greater than or less than a threshold. Figure 2 illustrates the saturation effect of this shape.

To get a better understanding of how to extend the definition of a consumer's quality preferences to derive the utility for the quality, let's take a closer look at the equations for these utility shapes. First, let's assume that the quality values for a quality Q are normalized such that $q \in \overline{Q}$ and $0 \leq q \leq 1$. For each utility shape we can derive equations constraining the curve's parameters. These constraining equations are such that, if the parameters are selected while satisfying them, we are able to derive a utility equation that approximates the consumer's utility for the quality.

5.1.2 Tradeoffs Among Qualities

We now consider a means to represent the consumer's preferences for tradeoffs among pairs of qualities.

DEFINITION 3 (PREFERENCE OPERATORS). *Following French [2, pp. 62–66], define for each consumer, the preference binary relations $\sim$ and $\succ$ for pairs of qualities:*

Indifference $A \sim B \subseteq \Phi \times \Phi$ *indicates that the consumer is indifferent between qualities A and B.*

Strict preference $A \succ B \subseteq \Phi \times \Phi$ *indicates that the consumer strictly prefers quality A over B.*

Preference $A \succsim B \subseteq \Phi \times \Phi$ *indicates that the consumer strictly prefers quality A over B, i.e., $A \succ B$, or that the consumer is indifferent between qualities A and B, i.e., $A \sim B$.*

The relation $\sim$ is reflexive, symmetric, and transitive, i.e., an equivalent relation. The relations formed by $\succ$ and $\succsim$ are both asymmetric and transitive, i.e., they are weak orders [2, p. 71].

Let's also assume that the consumer has a tradeoff preference for any two qualities of a service. That is, for any two qualities $Q_m, Q_n \in \Phi_s$ for a service either $Q_m \succ Q_n$ or $Q_n \succ Q_m$ or $Q_m \sim Q_n$. This is also stated in Axiom 1.

AXIOM 1. *For any service consumer of service s:*
$\forall Q_m, Q_n \in \Phi_s \Rightarrow (Q_m \succ Q_n) \vee (Q_n \succ Q_m) \vee (Q_m \sim Q_n)$
where $\Phi_s \subseteq \Phi_d$, $\Phi_d \subseteq \Phi$, and d is the domain of s.

Axiom 1 represents an ordering of the qualities for the consumer. This follows since $\succ$ is a strict order of the qualities and $\sim$ separates the set of qualities into disjoint sets of mutually indifferent qualities.

5.2 Quality Relationships

We consider two kinds of relationships between pairs of qualities: ontological and statistical.

- Ontological quality relationships are part of the QoS ontology and decided by the domain experts [6].
- Statistical quality relationships are inferred (by the agents) from the collected quality data.

DEFINITION 4 (ONTOLOGICAL RELATIONSHIPS). *Let Q be a service quality and let Φ be the set of all service qualities. $\Phi_d \subseteq \Phi$ is the set of all qualities for the service domain d of s and $\Phi_s \subseteq \Phi_d$ is the set of qualities applicable to a particular service s.*
Let $Q_{Rel} = \{Opposite(O), Parallel(P)\} \times \{Weak(W), Mild(M), Strong(S)\}$ be the set of semantic quality relationships.
We denote $\gamma : \Phi \times \Phi \mapsto Q_{Rel}$ as the relation representing the semantic relationship between pairs of qualities for a service domain d.

To help corroborate the ontological quality relationships provided by γ, we define the statistical quality relationship ρ as the correlation of the collected quality values. Let $\overline{Q_a} = \{q_l\}_{l=0}^{n}$ and $\overline{Q_b} = \{k_m\}_{m=0}^{n}$ be sets of values obtained for qualities Q_a and Q_b of an implementation of service s.

DEFINITION 5 (STATISTICAL RELATIONSHIPS). *The statistical relationship between qualities Q_a and Q_b (as measured) is given by the correlation between between the values as ordered in the sets $\overline{Q_a}$ and $\overline{Q_b}$ [8]. Thus $-1 \leq \rho(Q_a, Q_b) \leq 1$.*

As a concrete example of quality relationships, let us consider the relationships of the following pairs of qualities {*Throughput (T)*, *Capacity (C)*} and {*Robustness (R)*, *Availability (A)*}. Since *Capacity* is a measure of the limits on a service implementation's performance, we would expect it to be oppositely related to *Throughput*. On the other hand, qualities such as *Robustness* and *Availability* should be positively correlated. As agents collect values for these qualities we expect the qualities' ontological and statistical relationships to be in agreement.

5.3 New Service Trust Function

Assuming that the consumer's utility function for a quality can be derived with the necessary elicited input. We now define an improved consumer preference for qualities.

DEFINITION 6 (PREFERENCES FOR QUALITIES). *We redefine a consumer's preferences for a quality Q as $\phi = \{(q_{min}, q_{pref}, q_{max}), u()\}$ where q_{min} is the minimum value favored for the quality Q by the consumer, q_{max} is the maximum, q_{pref} is the preferred value, where $q_{min} \leq q_{pref} \leq q_{max}$, and $u() : \overline{Q} \mapsto [0, 1]$ is the consumer's utility for quality Q.*

Let us consider how to represent a consumer's service preferences using the new preferences for qualities as well as how to capture and incorporate the consumer's preferences for tradeoffs between pairs of qualities.

Definition 7 (Preferences for a Service). *Let $\Phi_c \subseteq \Phi_d$ be the set of qualities for which the consumer c of service s holds some preferences and where Φ_d is the set of all qualities for the service domain d of s.*
Let $\Phi_\sim \subseteq \Phi_c$ be the set of qualities for which the consumer is indifferent, that is, the consumer has no tradeoff preferences among them. Let $\Phi_\succ \subseteq \Phi_c$ be the set of qualities where the consumer holds tradeoff preferences. Clearly, $\Phi_c = \Phi_\sim \cup \Phi_\succ$.
We define the consumer's service preferences as $Pref = (\vec{\phi}, \vec{Q})$ where $\vec{\phi} = \langle \phi_1, \ldots, \phi_n \rangle$, in which $n = |\Phi_c|$, is an ordering of the quality policies ϕ; and $\vec{Q} = \langle Q_k \rangle_{k=1}^{|\Phi_c|}$ is a corresponding order for the qualities $Q_k \in \Phi_c$.

By definition of the operators $\sim$ and $\succ$ the following holds: $\forall Q_x, Q_y \in \Phi_\sim \Rightarrow (Q_x \sim Q_y)$. And $\forall Q_v, Q_w \in \Phi_\succ \Rightarrow (Q_v \succ Q_w) \vee (Q_w \succ Q_v)$. It also follows that using the operators $\succ$ and $\sim$, we can derive an ordering of the qualities for which the consumer has $\succsim$ tradeoff preference, i.e., $\vec{Q} = \langle Q_l \rangle_{l=1}^{m}$ such that $Q_1 \succsim Q_2 \ldots \succsim Q_m$ where $m = |\Phi_c|$.

Using the consumer's preferences for a service we can formulate a *trust* function. This function provides a solution to the selection problem of Definition 1 while taking into account the consumer's true preferences for qualities.

We achieve a trust assignment to a service implementation by matching the consumer's quality preferences to the service implementation provider advertisement and assigning a degree to the match. The matching degree takes into account the quality advertisements, preferences, and reputation.

Since the quality advertisements and preferences are defined as points on the quality line of Q, we can calculate the moment of these points with respect to the $\pi_{preferred}$ of the consumer preferences. In essence, the closer the advertised values and reputation are to the preferred value, the greater the degree of match (and of the resulting trust). Generally, Equation 1 shows the second moment of a vector $\vec{x} = \langle x_1, x_2, \ldots, x_n \rangle$ about some point a.

$$moment(\vec{x}, a) = \frac{1}{n-1} \sum_{i=1}^{n} (a - x_i)^2 \qquad (1)$$

We formulate the consumer's trust assignment for an implementation using Equation 1. However, since we want to match service implementations whose advertisement match the need of a service consumer, we begin by defining a matching operator between quality preferences and advertisements.

Definition 8 (Preference Matching Operator). *For each $Q \in \Phi_d$ let $\alpha_Q = (\alpha_{min}, \alpha_{typical}, \alpha_{max})$ is the advertisement of provider p of service implementation i for quality Q and $\pi_Q = (\pi_{min}, \pi_{preferred}, \pi_{max})$ be the consumer's preferences for quality Q.*
Let $Q_{min} = \min(\alpha_{min}, \pi_{min})$ and $Q_{max} = \max(\alpha_{max}, \pi_{max})$.
Let $\vec{Q_i} = \langle Q_{min}, \alpha_{typical}, \pi_{preferred}, Q_{max}, R_Q^{(i)} \rangle$.
We define the preference matching operator $\rhd$ for Q as:

$$\phi \rhd \varphi = \begin{cases} (\pi_{max} \le \alpha_{max}) \wedge (\pi_{preferred} \ge \alpha_{min}) \wedge \\ (\pi_{preferred} \le \alpha_{max}) \quad \text{if } dir(Q) = \uparrow, \text{ and} \\ \\ (\pi_{min} \le \varphi_{min}) \wedge (\pi_{preferred} \le \alpha_{min}) \wedge \\ (\pi_{preferred} \ge \alpha_{max}) \quad \text{when } dir(Q) = \downarrow \end{cases}$$

Where $dir(Q) : \Phi \mapsto \{\uparrow, \downarrow\}$ is associated with each quality Q, such that $dir(Q) = \uparrow$ indicates that the quality Q is directionally increasing which means that higher values for Q are generally preferred by service consumers. And $dir(Q) = \downarrow$ indicates that lower values are generally preferred.

Using the $\rhd$ operator we can derive the *trust* function of Definition 1 as follows.

Definition 9 (Service Trust Function).

$$\begin{aligned} qTrust(\vec{Q_i}, q_{preferred}) &= moment(\vec{Q_i}, q_{preferred})^{-\frac{1}{2}} \\ &\quad \text{where } moment(\vec{Q_i}, q_{preferred}) \neq 0 \\ serviceTrust(i) &= \sum_{\substack{Q \in \Phi_d, \\ \phi \rhd_Q \varphi}} qTrust(Q_i, q_{preferred}) \\ trust(i_p, c) &= serviceTrust(i_p) \end{aligned}$$

We now need a means to determine the importance of a quality Q_k in terms of its relationships with other qualities in $\vec{Q}$. We do this by finding an average (ontological and statistical) relationship value of Q_k when it is paired with all qualities that are less preferred (i.e., $Q_{k+1}, \ldots, Q_n$) and taking into account the quality compatibility of each pair.

Definition 10 (Quality Compatibility Operator). *We define two qualities $Q_x, Q_y \in \Phi$ as compatible if and only if they are directionally equivalent (See Section 4).*
The operator $\bowtie$ represents quality compatibility such that $Q_x \bowtie Q_y : \Phi \times \Phi \mapsto \{1, -1\}$ and defined as:

$$Q_x \bowtie Q_y = \begin{cases} 1 & \text{if } dir(Q_x) = dir(Q_y), \text{ and} \\ -1 & \text{otherwise} \end{cases}$$

Using the quality operator $\bowtie$ we can define an average quality relationship with the less preferred qualities in such a way that as we pair qualities to determine the relationship's value, we reinforce that value if the pair is compatible and negate it otherwise.

Definition 11 (Average Quality Relationship).

$$\varrho(Q_j) = \frac{1}{n-j} \sum_{m=j+1}^{n} \rho(Q_j, Q_m) \times (Q_j \bowtie Q_m) \text{ with } j \neq n$$

where ρ is the statistical quality relationship (Definition 5) and $\bowtie$ is the quality compatibility operator.
$\varrho(Q_j)$ represents the average relationship of the quality Q_j with the remaining qualities per the $\vec{Q}$ ordering.

We now derive a representation for the function *trust* that takes into account the consumer's new preferences for qualities, the consumer's utility for a quality, and the consumer's tradeoff preferences for qualities. Since we have an ordering for the qualities with $\vec{Q}$, we can derive the new *trust* function such that the first quality has more significance to the total value than the second quality, and so on.

Definition 12 (New Service Trust Function).

$$trust(i_p) = \frac{1}{n} \sum_{\substack{Q \in \Phi_c \wedge (\phi \triangleright_Q \varphi) \\ j=0}}^{n} w_j \times qTrust(Q_j, q_{pref}) \times [1 + \varrho(Q_j)]$$

where $i_p \in I_s$ is an implementation of service s by provider $p \in P$, $w_j \in \mathcal{R}$ is the weight the consumer associates with the quality Q_j. Assuming the constraint that the weights are in agreement with the consumer's preferences for the qualities, that is, $\forall Q_j, Q_{j+1} \in \vec{Q} | Q_j \succsim Q_{j+1} \Rightarrow w_j \geq w_{j+1}$. For correctness, the sequence of weights: $w_1, w_2, \ldots, w_{n-1}, w_n$ for n qualities can simply chosen as the sequence: $1, \frac{1}{2}, \ldots, \frac{1}{n-1}, \frac{1}{n}$.

The equation in Definition 12 represents the weighted sum of *qTrust* values for the qualities in Φ_c; however, for each quality $Q_j \in \vec{Q}$ we adjust its impact on the total trust value with the average relationship of that quality with all other qualities for which this quality is strictly preferred. The point of the adjustment is to give more impact to qualities that are compatible and on average correlate positively with the remaining qualities while reducing the impact otherwise.

6. EVALUATION

We have one provider of a loan service; however, we create multiple implementations to enable us to simultaneously examine specific relationships among qualities. Services in this domain expose the common qualities from a QoS middle ontology [6]; however, in addition, they have domain-specific qualities.

To simplify the *LoanService* implementations and the overall experiment, we consider only the following qualities:

- *LoanFee (LF)*. The fee that the consumer pays to the loan service upon approval and purchase of a loan.
- *InterestRate (IR)*. The interest rate that the consumer pays for the loan.
- *ApprovalRate (AR)*. The rate of approval of this loan service. That is, how often does the service approves a loan request versus denying the request?

Table 1: *QRel* for the *LoanFinancing* domain.

Quality pairs	Relationship	Correlation ρ
{ *LF*, *IR* }	Mild-Parallel (MP)	$\rho \simeq 0.50$
{ *LF*, *AR* }	Weak-Parallel (WP)	$\rho \simeq 0^+$
{ *IR*, *AR* }	Weak-Parallel (WP)	$\rho \simeq 0^+$

Table 1 show the initial QoS relationships among pairs of the qualities that we consider in this experiment. Table 2 shows the secondary parameters used for the simulations. The service doping names are based on the strength (Strong (S), Mild (M), or Weak (W)) and direction (Parallel (P) or Opposite (O)) of the relationship. For all simulations the consumer's preferences for qualities is: $\{LF \succ IR \succ AR\}$.

For this experiment we use one consumer and a set of five implementations from the same provider. All implementations expose the same advertisement, which remains constant throughout the simulations.

The results obtained are presented as service selection graphs (X-axis is execution sequence and Y-axis is selected service number).

Table 2: Secondary parameters of simulations.

Sim #	Weights	*LF*, *IR* service doping
0	$\{1.00, 0.80, 0.70\}$	NA
1	$\{1.00, 0.80, 0.70\}$	{ *SO*, *SO*, *SO*, *SO*, *SP* }
2	$\{1.00, 0.80, 0.70\}$	{ *MO*, *MO*, *MO*, *MO*, *SP* }
3	$\{1.00, 0.80, 0.70\}$	{ *SO*, *SO*, *MO*, *MO*, *SP* }
4, 5	$\{1, \frac{1}{2}, \frac{1}{3}\}$	{ *WP*, *WP*, *MP*, *MP*, *SP* }

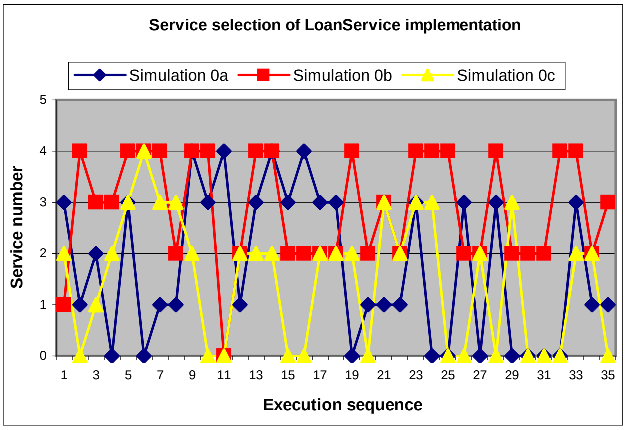

Figure 3: Simulation 0: simple preferences.

Simulation 0: Baseline service selection.
Figure 3 shows the results when the *LoanServiceAgent* does not take into account quality relationships, the consumer's complex preferences, and the augmented trust calculation, described in Section 5. The agent cannot decide on one implementation since they advertise the same quality policy and are identically implemented.

Simulations 1, 2, and 3: Selection for varying ρ.
Figure 4 shows the results for service selection when the service implementations are doped (artificially modified) to vary the quality relationships. The doping is according to Table 2 and affects qualities *LF* and *IR*. Since the consumer's quality preferences is constant:

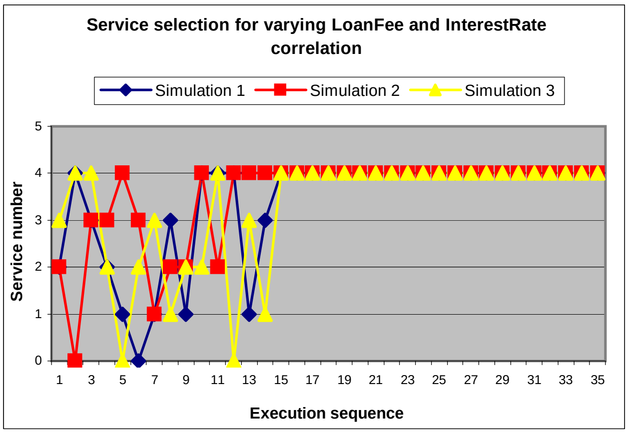

Figure 4: Simulations 1, 2, 3: varying *QRel*.

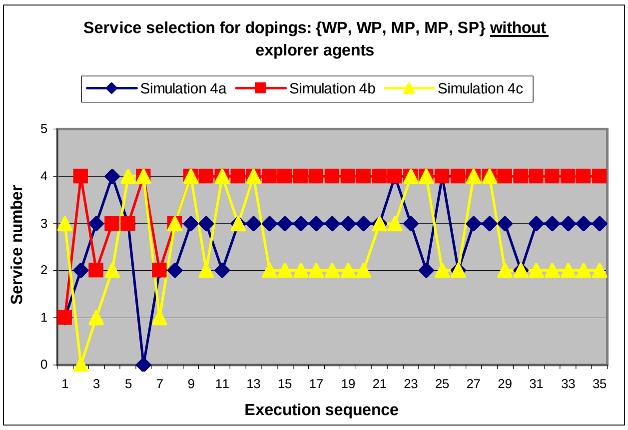

Figure 5: Simulation 4: no explorers.

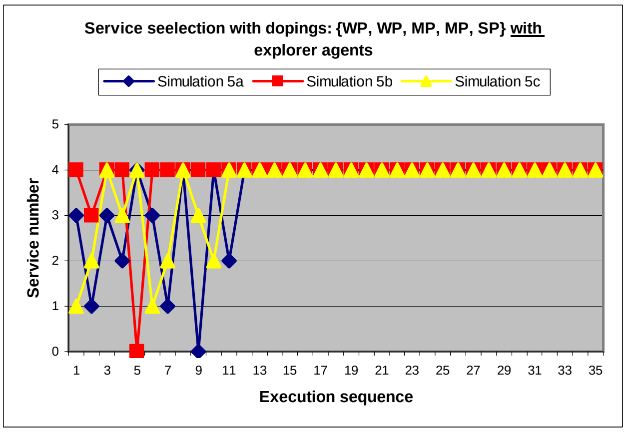

Figure 6: Simulation 5: with explorers.

$\{LF \succ IR \succ AR\}$ and since LF and IR are directionally compatible—a consumer generally prefers lower values of both qualities—we know that implementation four is the 'best' choice because it has SP relationship for LF and IR. As expected, the results show that for each simulation, the consumer eventually converges to the fourth service implementation.

Simulation 4 and 5: Effect of explorer agents. In this final set of simulations, we use the dopings: $\{WP,\ WP,\ MP,\ MP,\ SP\}$, which means that all services have relationships (LF, IR) that are directionally parallel but of different strengths. Figure 5 shows the results for three separate runs of this simulation. Notice that the agent is not able to always find the 'best' implementation since it sometimes gets stuck on a local maximum (i.e., an implementation with doping MP). Since the agent is selecting the MP service it is building up its reputation, which makes it harder for the agent to find the best implementation.

A possible remedy for the local maximum problem is to use explorer agents that select all services without discrimination and help them build their reputations, independently of the consumer agent's selection. We added three explorer agents that run a round-robin selection algorithm. The explorer agents run a script similar to the one used by the *LoanServiceAgent*.

Figure 6 show the results for this experiments. We included three runs of the simulations, each showing consistent convergence to the 'best' service implementation (with doping SP). Notice also that the convergence is faster than in Simulations 1, 2, and 3. We attribute the fast convergence to the explorer agents since they help all implementations build up their reputations and, therefore, indirectly help the consumer agent to quickly find the SP implementation.

7. RELATED WORK

Service selection and binding approaches fall into two primary categories: design-time and runtime. In design-time selection and binding, the application designer or architect use service registries coupled with service descriptions to select and test binding to a service. Nonfunctional characteristics are considered during trial and error tests of the selected services. Newer techniques using richer semantic descriptions of services can help in the discovery of service interfaces. OWL-S [10] is an example of a rich service ontology used for semantic service discovery.

The other category, of most concern to our work, is runtime service selection and binding. In this case, the service interface is already discovered. At runtime, the service implementations are discovered, selected, and bound to—all based on QoS requirements, models and metrics, and middleware. Key models are those of the W3C [11] and UML [1]. Brokering and middleware approaches that use QoS include works by Ran [9] and by Wohlstadter et al. [12].

Using QoS for service selection is also proposed by Ran [9], Kalepu et al. [4] and Zeng et al. [14]; however, while these work fail to address selection adequately for open environments because they do not consider trust, reputation, and a decentralized multiagent architecture as we are proposing. Wohlstadter et al. propose a policy language for advertising the QoS needs of clients and to allow the middleware to match servers to clients [12]. However, their work does not address a complete conceptualization of nonfunctional attributes for Web services. Further, Wohlstadter et al.'s matchmaking techniques lack support to enable dynamic evolution of the QoS exposed by the services and does not consider relationships that exist between qualities.

The literature on trust neither addresses QoS directly nor the applicability of trust for dynamic and autonomic service selection. Huynh et al. give a framework for trust determination in open systems, but do not use QoS conceptualizations to determine the needs of the truster [3]. Zacharia and Maes give a general model for reputations that we extend into dynamic service selection [13].

Finally, our agents are decision-theoretic in design. Their decision-making algorithm, used to make selection decisions, applies the works by Keeney and Raiffa [5] in decision making with multiple objectives, extending it in terms of ontologies and computing with quality relationships.

8. DISCUSSION

Solutions to the dynamic service selection problem are crucial for realizing the potential of Web services. After a mathematical formulation of the problem, we proposed an initial solution built on a trust model of services that directly take into account the qualities advertised by service providers and the quality preferences of consumers. Considering more detailed consumer preferences for qualities, we presented an updated trust model where the preferences for qualities take into account the shape of the consumer's utility curve for a single quality, the consumer's preferences for tradeoffs among pairs of qualities, and the relationships between pairs of qualities.

An area of future work is to complement our global view of trust with one that is local to each agent. Essentially, each agent would keep track of its past interactions with service providers and service instances and would can use the resulting computed trust value to complement the global view.

A natural follow-on to this preference model is to consider the consumer's utility for multiple (greater than two) qualities, the tradeoffs between multiple qualities, and making explicit the relationships between multiple qualities. A promising approach is to apply multi-objective utility and decision theory, as studied by Keeney and Raiffa [5]. Finally, our current model for trust does not consider how truthful providers are in their QoS advertisements and does not include a model to capture transitive trust. These could be added to our model by including an *honesty* notion that compares a provider's advertisements to its reputation for qualities and by introducing *endorsements* between providers and a consumer's list of trusted providers.

Finally, an analytical model that complements our problem formulation and empirical evaluation would give insight into the system and also help derive mechanisms to give incentives discouraging service consumers from being untruthful and allow them to share.

9. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their comments.

10. REFERENCES

[1] J. Aagedal, M. de Miguel, E. Fafournoux, M. Lund, and K. Stolen. UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms. TR 2004-06-01, OMG, 2004.

[2] S. French. *Decision Theory: An Introduction to the Mathematics of Rationality*. Wiley, 1986.

[3] D. Huynh, N. Jennings, and N. Shadbolt. Developing an Integrated Trust and Reputation Model for Open Multi-Agent Systems. *AAMAS)*, 2004.

[4] S. Kalepu, S. Krishnaswamy, and S. Loke. Verity: A QoS Metric for Selecting Web Services and Providers. *Proc. 4th Intl. Conf. Web Information Systems Eng. Workshops*, pp. 131–139, 2003.

[5] R. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Wiley, 1976.

[6] E. Maximilien and M. Singh. A Framework and Ontology for Dynamic Web Services Selection. *IEEE Internet Comp.*, 8(5):84–93, 2004.

[7] E. Maximilien and M. Singh. Toward Autonomic Web Services Trust and Selection. *Proc. of 2nd Intl. Conf. Service Oriented Comp.*, pp. 212–221, 2004.

[8] D. Montgomery, G. Runger, and N. Hubele. *Engineering Statistics*. Wiley, 3rd ed, 2004.

[9] S. Ran. A Framework for Discovering Web Services with Desired Quality of Service Attributes. In L.-J. Zhang, ed, *Proc. Intl. Conf. Web Services*, pp. 208–213, 2003.

[10] K. Sycara, M. Paolucci, A. Ankolekar, and N. Srinivasan. Automated Discovery, Interaction, and Composition of Semantic Web Services. *J. Web Semantics*, 1(1):27–46, 2003.

[11] W3C. QoS for Web Services: Requirements and Possible Approaches, 2003. Note.

[12] E. Wohlstadter, S. Tai, T. Mikalsen, I. Rouvellou, and P. Devanbu. GlueQoS: Middleware to Sweeten Quality-of-Service Policy Interactions. In *Proc. 26th Intl. Conf. Soft. Eng.*, pp. 189–199, 2004.

[13] G. Zacharia and P. Maes. Trust Management Through Reputation Mechanisms. *Appl. Art. Intel.*, 14:881–907, 2000.

[14] L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. QoS-Aware Middleware for Web Services Composition. *IEEE Transactions on Soft. Eng.*, 30(5):311–327, 2004.