ASSESSING SERVICE PROPERTIES WITH REGARD TO A REQUESTED QOS: THE SERVICE METRIC

Claudia Linnhoff-Popien[1] and Dirk Thißen[2]

[1]*Ludwig Maximilian University München, Institute for Computer Science*
Oettingenstr. 67, 80538 München, Germany
E-Mail: linnhoff@informatik.uni-muenchen.de
Phone: +49-89-21782149, Fax: +49-89-21782147

[2]*Aachen University of Technology, Department of Computer Science, Informatik IV*
Ahornstr. 55, 52056 Aachen, Germany
E-Mail: thissen@i4.informatik.rwth-aachen.de
Phone: +49-241-8021410, Fax: +49-241-8888220

Abstract

The growth of computer networks and the increasing transmission capacity enables a rising number of service offers in distributed systems. One of the CORBAservices is the trading service, which supports clients searching for suitable services.

This paper presents the idea of a service metric which can be implemented in an evaluation component for a CORBA trader. A service distance function is used to select an optimal service offer from a trader's database. It computes the distance between a service request and a service offer with regard to their service properties. The service with the minimal distance to the service request represents the best offer. Hence, existing methods for distance computation between vectors are used, and a set of rules for computing the distance between service properties is developed. The resulting evaluation procedure is implemented in a CORBA trader using the distributed platform Orbix. The implementation is compared with the classic evaluation mechanism of the trader. Furthermore, the various methods for distance computation are compared.

Introduction

A large distributed system offers many services. To support a client searching for a particular service a trader can be used. This trader contains specifications of services by a service type and service property values. If a client requests a service, the trader

P. Ciancarini et al. (eds.), *Formal Methods for Open Object-Based Distributed Systems*

selects a suited one. The problem in this mechanism is that the process of selecting a service only matches the client's request against the service property values. There is no possibility to take into consideration the quality of service aspects as defined in the quality of service framework [7]. Hence, the common trading mechanism causes two problems. First, there is only a limited chance to identify an order of precedence of the service offers found. Accordingly, it may well happen that the trader selects a less suitable service although more suitable services are registered in the trader's service directory. The second problem is that no service is selected if all service offers in the trader's domain have a small deviation from the importer's specification. Nevertheless, a slightly deviating service could satisfy the importer.

This paper introduces a modification of a trader's evaluation procedure, which allows a client to include quality aspects into its request. The trader now uses a set of service distance functions to calculate the distance between the service request and each service offer to produce a ranking on the given service offers. Therefore, a service is viewed as a vector of properties, and a service metric is introduced, which uses analytic methods to calculate distances between such vectors. With respect to service quality a property can consist of several values with certain roles. Thus, to calculate a distance between these components, it was necessary to develop a new set of rules. The service with minimal distance to the requested service is regarded as optimal.

In the following section, the necessary modifications of a service property's description for considering quality of service is presented. Furthermore, the concept of a service metric and its integration into the service trading process are explained. Subsequently, some implementation aspects of the corresponding evaluation component are briefly described, and an evaluation of this component is given.

Service trading using service distance functions

To explain the purpose of service distance functions, a short introduction to service notations and service trading [6] is given first. The service distance functions are then introduced and the enhanced service selection process in the trader's service directory is explained.

Service trading and quality of service

A service offered by an object can be described by a *service type* and several non-computational aspects called *service properties*. Different services of the same service type may differ in their service properties. A service property is described as a (name, value) pair. A *trader* is a service selecting other services. Thus, a trader needs a *service directory*, which contains descriptions of a number of services in terms of service type and service properties. A service description included in such a directory is called a *service offer*. An *exporter*, i.e. a server that wants to provide a service, can register this service with the service directory. A client using a trader to search for a service is called *importer*. It requests a service from the trader by specifying the service type and constraints on some service properties. The selection of suitable services is made by the trader by matching service type and service properties with every service offer. The importer receives an interface identifier, which can be used to directly contact the server that has exported the selected service.

Service Request: $\mathbf{r} = (r_1, ..., r_n)$ with
Service Request Property r_i :
r_i= { Target t
Upper Bound u
Lower Bound l
Preference p
}

Service Offer $\mathbf{o} = (o_1, ..., o_m)$ with
Service Offer Property o_i :
o_i = { Upper Bound u
Lower Bound l
}

Figure 1. Service property structures

The trading mechanism described only tolerates single values or value ranges for each service property. These values are compared with the corresponding values of the service offer. A single value is either matched or it is not. This concept is insufficient for many applications. Many services, like video conferencing and interactive applications, need a consideration of quality aspects. The *quality of service* could be taken into account by formulating quality characteristics as service properties. Quality characteristics include throughput, transmission delay, or availability. Now, an importer may need to specify more than one value, namely a *target*, a *lower bound* and an *upper bound* for such a characteristic to define its requirements and limits, see figure 1. All these values are components of a quality of service parameter, so they are all needed to express quality [7]. Furthermore, the importer must be allowed to specify the *preference* for every characteristic, to inform the trader about priority characteristics. In a similar way, an exporter has to specify a lower bound and an upper bound for each service property to express its quality capabilities, see [4]. Because of the resulting complex structures of service properties, 'normal' matching no longer can be used. All service properties can be grouped in a service request property vector in the case of an importer, and in a service offer property vector in the case of an exporter.

The service metric

Two steps are necessary to compute the distance between service vectors, see figure 2. After receiving a service request (1), the service metric, which extends the trader's computation layer, calculates a ranking on service offers in two steps. The first step is to compute the distance between the components of both vectors, the service property records (2). The result is a vector of differences. In the second step (3), these differences must be combined to give the distance between vectors. The result of this step is a sorted list of service offers (4), which can be passed to the importer (5).

Service property record distance. As described before, in our example a service request property with quality consideration consists of four values with the roles target, upper bound, lower bound and preference, respectively. A service offer property consists of a lower bound and an upper bound. As it is impossible to compute a simple difference between both structures, it is necessary to draw up rules for each possible combination of values.

The importer is allowed to specify any combination of target and bounds. To simplify the evaluation, either both bounds or no bounds are defined in the specifying

process. Undefined components have the value 'undefined'. Likewise, the exporter can specify either no bound or both bounds. Service offer vector and service request vector do not need to have equal size and contain equal properties, so o_i only means the property with the same name as r_i. This paper does not give the full set of rules, which can be found in [10].

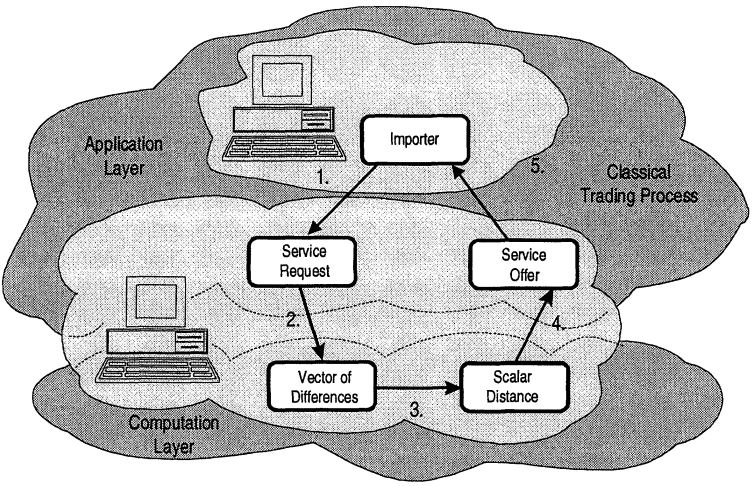

Figure 2. The service import process using distance functions

As an example, if the target as well as bounds are given by the importer, the distance is zero, if the target is within the provider's bounds. If there is no overlap between the importer's and the provider's interval, the distance is infinite. Otherwise, the distance is the difference between importer's target and the provider's bound that is within the importer's interval:

$$\begin{aligned}
&\underline{\text{if}}\ r_i.t \neq \perp\ \underline{\text{and}}\ r_i.u \neq \perp\ \underline{\text{and}}\ o_i.u \neq \perp\ \underline{\text{then}}\\
&\quad \underline{\text{if}}\ o_i.l \leq r_i.t \leq o_i.u\\
&\qquad \underline{\text{then}}\ d_c(r_i, o_i) = 0\\
&\quad \underline{\text{else if}}\ [r_i.l, r_i.u] \cap [o_i.l, o_i.u] = \emptyset\\
&\qquad \underline{\text{then}}\ d_c(r_i, o_i) = \infty\\
&\quad \underline{\text{else if}}\ r_i.t < o_i.l \leq r_i.u\\
&\qquad \underline{\text{then}}\ d_c(r_i, o_i) = o_i.l - r_i.t\\
&\qquad \underline{\text{else}}\ d_c(r_i, o_i) = r_i.t - o_i.u
\end{aligned}$$

Following this computation, the resulting difference d_c must be weighted using the importer-specified weights to consider the importance of a property, for example:

$$d_{wc}(r_i, o_i) = r_i.p * d_c(r_i, o_i)$$

The whole set of rules allows to compute the distances for all components of the service request vector. The resulting differences could be put together in a difference vector $\mathbf{d} = (d_1, ..., d_n)$ with $d_i = d_{wc}(r_i, o_i)$ for i = 1, ..., n, if the service request contains n service properties. In the next step all these values d_i are to be combined to compute the difference between the service vectors.

Service property vector distance. The idea of a vector distance function d_V is to compute a distance between vectors by combining the individual component differences. There are a few metrics which can be employed on such vectors. In general, an analytic metric for $n > 0$ is defined as follows:

$$d_v((x_1,\ldots,x_m),(y_1,\ldots,y_m)) := \sqrt[n]{\sum_{i=1}^{m} d_{wc}(x_i, y_i)^n}$$

In practice, the relevant cases are n = 1, 2 and infinity. In these cases, the metrics are called Manhattan metric, Euclidean metric and maximum metric, respectively. The optimal service offer can be found by minimizing the distance d_v. In addition to the mathematical functions to compute the distance between vectors, other methods to find an optimal service include:

- Fuzzy set theory [11],
- Analytic hierarchy process (AHP) [9], [2], and
- Additive and multiplicative model [1].

A description of these methods can be found in [10]. Because of the easy implementation possibilities of all methods except the analytic hierarchy process, the best solution is an integrated distance computation mechanism in one evaluation component with a choice of the distance function for the importer. In this case, the user could choose a special function to express a special meaning of "optimal".

Implementation and assessment of the evaluation component

The evaluation component was implemented, and integrated into an in-house implementation of a CORBA trader [12], [5] of the distributed platform Orbix. The importer must specify the service request as before, but additionally it can specify more values of a service property. Furthermore, the importer must choose a service distance function and a weighting mechanism. Different importers could have different ideas of an optimal service. Therefore, every combination of a distance function with both multiplicative and exponential weightings is possible in the evaluation component. Fuzzy set theory optimization and the additive model can be interpreted as special cases of maximum metric and Manhattan metric, respectively. Thus, these types do not need to be considered. To be consistent with the trading standard, the service properties cannot be transmitted to the trader as a vector, but must be transformed into the trader's format. Within the trader, this transformation must be reverted. An importer specifies two lists with its service property descriptions. The *constraint* list consists of the targets and bounds of all the importer's service properties, the preferences are stored in the *preference* list. The trader transforms these lists into the service request property vector, i.e. an Orbix sequence. The service properties are sorted by lexical order. Likewise, the trader sorts the service properties specified by an exporter into a service offer property vector. The evaluation procedure of the trader was enhanced to compute the distance between the vectors as described in the previous section. In the first step a vector is computed, which contains all component distances. The chosen distance function is executed on this vector and the corresponding property preferences. All suitable service offers are collected in a list which is sorted by the computed distance. So, the first entry is the optimal service.

To assess the evaluation component, a comparison with the 'classic' trader, i.e. a trader only matching a service request against the service properties of a service offer, will be necessary. Furthermore, the different distance functions should be compared with each other. In the following, all service offers have the same service type as only the service properties are of interest. The service property values are chosen by chance to generate a random environment. The service request has the same service type and a subset of the properties of the service offers.

The dependence of the selection process on the number of service offers in the trader's directory was presented in [4] and [8]. In the following, the influence of the number of service properties on the evaluation time is discussed, as well as changing evaluation results caused by varying the distance computation process.

In one measurement, the influence of the number of service properties specified in the importer's service request was analyzed, measuring time for the evaluation process as a function of the number of service request properties. Hence, 20 service offers with 50 service properties each were exported to the trader. The service import took place with a varying number of service properties. In figure 3, the dependency of the evaluation time on the importer's number of service properties can be seen. The distance functions only cause a slightly constant ascend in the evaluation time caused by the constant growth of the property vector's length.

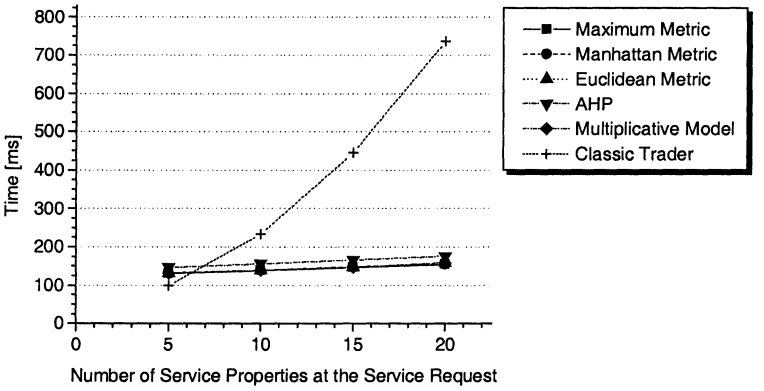

Figure 3. Duration of the whole evaluation in dependency form the number of service properties of the service request

For a small number of service request properties, the classic trader is faster than the trader working with distance functions. But with an increasing number of service request properties, the behaviour of the trader becomes rapidly bad in comparison to the enhanced trader. This is due to the implementation of the trader's evaluation procedures. In the classic trader, the service offer property list is searched for each service request property. With an increasing number of service request properties, computation time increases as well. The enhanced trader takes advantage of the lexical order of the property vectors. As a side effect of this order, only a single list search has to be done.

A similar measurement was made to analyze the dependency of the evaluation time on the number of the exporter's properties. All evaluation strategies show a

proportionality between the evaluation time and the number of service properties, as the whole offer property vector is searched.

For an evaluation of the service distance functions not only the evaluation time is of interest, but also the changes in the evaluation results due to different evaluation strategies. One possibility to vary the evaluation startegies is to choose between multiplicative and exponential consideration of the preferences. It turned out that the multiplicative use of preferences yields a good distinction between the metrics, whereas the multiplicative model rates the most suitable service offers equally. On the other hand, the exponential use of the preferences causes smaller differences between the metrics, but a more detailed differentiation between the more suitable service offers in the case of using the multiplicative model.

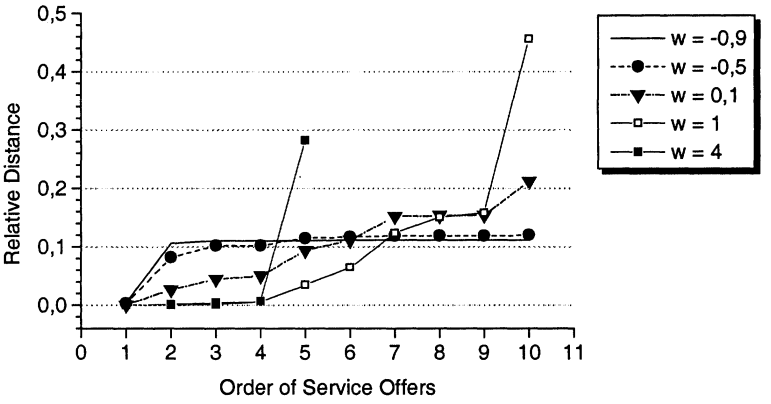

Figure 4. Relative distances to all service offers for the multiplicative model with varying control parameter w

Another possibility considered to vary the evaluation result is given by the multiplicative model. It has an additional parameter *w*, which can be used to control the evaluation result. Hitherto, *w* was set to a default value of 1. In the following, different values are examined, see figure 4. Note, that not the real distances between service offers and a service request are given in the measurements, but normalized values. This is because the distance functions compute values of different ranges. To compare them it is necessary to use a normalized distance, the relative distance. The service import was done as before, with multiplicative consideration of the preferences. With an increasing value of *w*, the differences between the more suitable service offers becomes smaller, and the differentiation between the less suitable service offers improves. But the ranking of the service offers is hardly influenced at all by the variation of *w*. Hence, the value of *w* is not of interest in the service selection process, but it may be used to vary the number of service offers with a relative distance to the service request below a given reference value.

The behaviour of the distance functions, especially the multiplicative model, leads to the idea of extending the import operation. The importer could specify a distance of the interval [0, 1]. Only those service offers with a relative distance lower than the importer's distance parameter are presented. Such a variation of the importer's operations is not possible with the classic trader.

Conclusions

This paper has presented the idea of a service metric to formulate and evaluate service properties with regard to quality. For that purpose services were viewed as vectors, which enabled us to compute the distances between a service request and each service offer. An optimal service offer has a minimal distance to the service request.

The result of this paper is an evaluation mechanism for distance computing between services. This set of rules is more flexible and more importer oriented than common selection mechanisms, because it on the one hand allows the importer to specify the quality of the required properties, on the other hand it allows to express its interpretation of an optimal service. As a result the importer is offered services that better suit its needs. As there are also no major disadvantages of the distance functions in terms of evaluation time, the evaluation component is suitable. Furthermore, an extended importer can be implemented using service distance functions.

One future task is to refine the set of rules, to cover for example the case, where a property specified by the importer is not available. Furthermore, our own trader implementation has to be enhanced. For more general predications about the usefulness of the service metric, a comparison with the OrbixTrader [3] should be made.

References

[1] Bernard, R.: *Decision-Aid and Decision-Making.* In: Bana e Costa, C.: Readings in Multiple Criteria Decision Aid; Springer-Verlag, 1990.

[2] Douligeris, C.; Pereira, I.: *A Telecommunications Quality Study Using the Analytic Hierarchy Process.* IEEE Journal on Selected Areas in Communications, Vol. 12, No. 2, pp. 241-50, Feb. 1994.

[3] IONA, *OrbixTrader Programmer's Guide and Reference.* IONA Technologies, Ltd., Release 1.1, 1998.

[4] Linnhoff-Popien, C.; Thißen, D.: *Integrating QoS Restrictions into the Process of Service Selection.* In: Campbell, A.; Nahrstedt, K.: Building QoS into Distributed Systems, Fifth International Workshop pn Quality of Servcie, New York, 1997.

[5] ISO/IEC DIS 13235, *ODP Trading function*, 1995.

[6] Popien, C.; Schürmann, G.; Weiß, K.-H.: *Distributed Processing in Open Systems* (in German). Teubner, 1996.

[7] ISO/IEC DIS 13236: *Quality of Service - Framework*, October, 1996.

[8] Reichl, P.; Linnhoff-Popien, C.; Thißen, D.: *Comprehension of user interests for a QoS-based service trading in CORBA* (in German). In: Zitterbart, M.: Kommunikation in Verteilten Systemen, Springer, 1997.

[9] Saaty, T.L.: *The Analytic Hierarchy Process.* McGraw-Hill, 1980.

[10] Thißen, D.: *QoS based Optimisation of Service Selection within an Orbix Trader* (in German). Diploma thesis at the Department of Computer Science IV at Aachen University of Technology, July 1996.

[11] Zimmermann, H.-J.: *Fuzzy Set Theory and its Applications*, 2nd edition. Kluwer Academic Publishers, 1991.

[12] Zlatintsis, S.: *Design and Valuation of a Trader Gateway between ANSAware and ORB systems* (in German). Diploma thesis at the Department of Computer Science IV at Aachen University of Technology, February 1996.