Role-based Service Description and Discovery

Alberto Fernández, Matteo Vasirani, César Cáceres, Sascha Ossowski

Artificial Intelligence Group, University Rey Juan Carlos,
Calle Tulipán s/n, 28933 Móstoles (Madrid), Spain
{alberto.fernandez,matteo.vasirani,cesar.caceres,sascha.ossowski}@urjc.es

Abstract. The ever-growing number of services on the WWW provides enormous business opportunities. Services can be automatically discovered and invoked, or even be dynamically composed from more simples ones. In this paper we concentrate on the problem of service discovery. Most current approaches base their search on inputs and outputs of the service. Some of them also take into account preconditions and effects, and other parameters that describe the service. We present a new approach that complement existing ones by considering the types of interactions that services can be used in. We present our proposal for a concrete application based on a real-world scenario for emergency assistance in the healthcare domain.

1. Introduction

Service-Oriented Computing [9], [10] is a software paradigm for distributed computing that is changing the way software applications are designed. Services are computational entities that can be described, published, discovered, orchestrated and invoked by other software entities. The Semantic Web Services Architectural framework [1] attempts to address the dynamic service discovery, service engagement, service process enactment and management, community support services, and quality of service.

Agent technology provides designers with an interaction-oriented way of designing open software systems [15]. There is a growing awareness that organisational models are fundamental to regulate open multi-agent systems so as to instil desired properties [23], [21], [19]. Service-oriented and multiagent systems are quite related [11]. Often, web service technology is used to support the interactions in multiagent systems [5].

In this paper we concentrate on the problem of dynamic service discovery in multiagent systems. Most current service discovery techniques [14], [17], [20] aim at web services and base their search on inputs and outputs of the service. Some of them also take into account preconditions, effects and other parameters that describe the service. However, agent-based service discovery mechanisms can also make use of the information provided by the organisational model underlying the multiagent system. Following this idea, we present a new approach that extends existing mechanisms by
considering the *types of interactions* that services can be used in. We have built a service matchmaker following this approach within the CASCOM project [2] [8]. To describe our proposal we use examples from a concrete application based on a real-world scenario for emergency assistance in the healthcare domain.

This paper is organised as follows. In section 2 we present a role-based interaction modelling approach for service descriptions and illustrate it in the aforementioned use case scenario. In section 3 we outline what organisational information is relevant for service descriptions in multiagent systems, and show how it can be represented in OWL-S. In section 4, the matchmaking algorithm is presented. Finally, section 5 summarises our proposal and points to future lines of work.

2. Role-based Interaction Approach to Service Description and Discovery

In order to improve both the efficiency and the usability of agent-based service-oriented architectures, we suggest exploiting common organisational concepts such as social roles and types of interactions to further characterise the context that certain semantic services can be used in. In this section we outline our role-based and interaction-centric modelling framework [18].

The CASCOM abstract architecture [3] conceives services to be delivered essentially by agents. In such an approach the agents usually act as mere wrappers for web services. The difference between a web service and a service provided by an agent boils down to a matter of interface: an agent can provide an implemented web service by a process of wrapping the service within an ACL interface in such a way that any agent can invoke its execution by sending the adequate (*request*) message.

However, agents are not only able to execute a service but can also engage in different types of interaction with that service. For example, consider a healthcare assistance scenario, similar to the one treated in the CASCOM project [4]: an agent providing a *second opinion* service should not only be able to provide a diagnostic; it may also be required to explain it, give more details, recommend a treatment, etc. This means that the service provider is supposed to engage in several different interactions during the provision of a service. Thus, if a physician or a patient needs one or more second opinions, they should look for agents that include those additional interaction capabilities around the “basic” *second opinion* service. In a certain sense, this approach is similar to the abstraction that an object makes by providing a set of methods to manipulate the data it encapsulates. In this case, the agent provides a set of interaction capabilities based on the service.

Most current OWL-S matchmakers [17], [14] only consider logical inference on service inputs and outputs in their matching algorithm. In the CASCOM project we use OWLS-MX [13], a hybrid semantic matchmaker that also uses information retrieval techniques for service matching. We have extended this matchmaker with role- and interaction-based matching techniques in order to adapt it to scenarios as the one outlined above.

In particular, the *efficiency* of the matchmaking process can be improved by previously filtering out those services that are incompatible in the terms of roles and
interactions. The precision of the matchmaking process can also be enhanced by including information regarding the roles and interactions. For instance, a diagnosis service may require symptoms and medical records as inputs and produce a report as output. However, the service functionality can be achieved either (i) by actually generating the report, (ii) by retrieving a previously done or (iii) by a brokering service to contact other (external) healthcare experts. As we will outline below, in all three cases the inputs and outputs are the same, but the role the service plays in the corresponding interactions is different.

2.1 Interaction Modelling

In order to develop role-based extensions to service discovery mechanisms we use a subset of the RICA organisational model described in [18] and [19]. Setting out from this basis, we first analyse different use cases of the application domain scenario. For each use case, we identify the types of social interaction as well as the roles (usually two) that take part in that interaction. The next step is an abstraction process in which the social (domain) roles/interactions are generalised into communicative roles/interactions.

In the sequel, we will illustrate our approach based on the second opinion use case in the healthcare domain. In this scenario, the patient (or the physician of a local emergency centre) can ask an external health professional for a diagnosis on the basis of the symptoms and the medical records of the patient, like exams and past diseases.

The “conversation” between the patient and the health professional can be modelled by a sequence of (communicative) actions between the two agents involved, as depicted in Fig. 1. The patient asks the health professional for an opinion, providing the symptoms and the medical records. If there is insufficient information,
the health professional requests additional information (possibly several times) and finally gives his advisement. If the provided diagnosis is not sufficiently clear, the patient can also solicit an explanation.

Starting from this conversation we can isolate 3 different interactions: (i) the second opinion exchange, which can comprise (ii) a detailed information exchange. When the second opinion exchange finishes, an explanation (iii) can occur.

Having identified the basic interactions, we can model the second opinion use case in a more precise way, as depicted in Fig. 2, and furthermore define the roles involved in the interactions. The result of this analysis is a basic ontology of roles. Fig. 3 shows an example, where the role SecondOpinionRequestee is generalized into a MedicalAdvisor role, which in turn is generalized into an Advisor role. Similarly, the SecondOpinion interaction can be generalized in a MedicalAdvisement interaction and then in an Advisement interaction, in which the Advisor informs the Advisee about his beliefs with the aim of persuading the Advisee of the goodness of these beliefs.
2.2 Role and Interaction Ontologies

From the use case scenarios, we have derived an ontology that contains a taxonomy of types of interactions, and a taxonomy of roles that take part in those interactions. Fig. 4 shows a UML class diagram representation of the interaction type ontology. There are two kinds of interactions, social interaction types and communicative interaction types. Social interaction types are domain interactions (Emergency Medical Assistance, Medical Advisement, etc.). Communicative interaction types are generic reusable interaction patterns. They constitute abstract communication interactions that can be instantiated to different scenarios. For instance, a medical advisement is a specialisation, in the medical domain, of the generic advisement interaction type. In the UML diagram generic (communicative) interaction types are represented as interfaces, whereas domain (social) interactions are represented by classes.

![Interaction type ontology](image)

Fig. 4. Interaction type ontology

The top concept of the ontology is the Communication, which represents the most generic type of interaction. Any type of interaction is a specialisation of this concept. The first level of specialisation is the ClosedActionPerforming interaction type, which represents any interaction that implies an action performing. The tag closed means the interaction has a fixed number of participants (usually two), unlike OpenActionPerforming representing interactions where that number is not predetermined (e.g. a call for proposals). Explanation, Advisement, Admission, Assistance, InformationExchange and Brokering are specialisations of
ClosedActionPerforming. For example, an information exchange is a particular case of action performing, where the informee asks the informer to execute the action of informing about certain fact. Similar analysis can be applied to the domain interaction types that specialise communicative interaction types into social interaction types. For instance the ArrivalNotification is a particular kind of notification interaction about the arrival of a patient to a hospital. New (generic and domain) interaction types can be incrementally be added to this taxonomy making it more complete and reusable.

In line with the interaction taxonomy, there is a role taxonomy which includes a hierarchy of the roles that participate in the interactions. For each concept in the interaction ontology there will be at least two participant concepts in the role ontology. For example, the informer and informee are concepts of the role ontology that participate in the information exchange interaction.

These ontologies, and especially their generic (communicative) part, will be used in the service description and matchmaking extensions.

3. Role-based Service Description

In this section we outline our proposal to integrate role-based information into service descriptions. We first describe a representation for role-based service advertisements and service requests, and then explain how this information can be included in the OWL-S service profiles, the service description language we use.

3.1 Service Advertisements

Role-based service descriptions comprise two kinds of information related to the interactions in which the service provider agent can engage:
1) the main role played in the interaction, e.g. the advisor role in the second opinion service;
2) a set of roles that may be necessary to be played by the requester for the correct accomplishment of the service. For instance, in an advisement interaction of a second opinion service, the provider may need to initiate an information exchange interaction in which it plays the informee role, and the requester plays the informer role. Necessary roles are given by a formula in disjunctive normal form, i.e. a disjunction of conjunctions of roles.

These two fields are repeated for each main role the service can play. We may graphically represent a service advertisement by a table with two rows, in which each column contains the main role (first row) and the necessary roles (second row).

Table 1. Abstract example of a role-based service advertisement

<table>
<thead>
<tr>
<th>Main Role</th>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Necessary roles</td>
<td>$(R_4 \wedge R_5) \vee R_6$</td>
<td>$(R_6 \wedge R_7 \wedge R_8) \vee (R_4 \wedge R_7)$</td>
<td></td>
</tr>
</tbody>
</table>
In this abstract example, the service provider agent can engage in three different types of interactions when providing the service. It can play role \( R_1 \), which requires the requester to be able to play both \( R_4 \) and \( R_5 \), or \( R_6 \). If this condition fails, then the provider will not be able to carry out the service properly. However, when the provider plays the role \( R_2 \), it requires the requester to play either roles \( R_6 \), \( R_7 \) and \( R_8 \), or otherwise roles \( R_4 \) and \( R_7 \). In the case of \( R_3 \), no role playing capabilities are mandatory for the requester.

### 3.2 Service Requests

In the case of a service requests, we consider that a query comprises two elements:

1) *Main roles* searched. Although one role will be enough in most cases, we allow for more complex search patterns in which the provider is able to play more than one role. As in the case of service advertisements, we require an expression in disjunctive normal form here.

2) A set of roles that define the *capabilities* of the requester. These are roles the requester is able to play. This information is important if the provider requires interaction capabilities from the requesters. For example, the requester of a second opinion can inform that it is able to provide information (informer) if needed.

Table 2 shows an abstract example of a query in which the requester searches a service provider able to play either \( R_2 \) or both \( R_4 \) and \( R_6 \) to provide the service. In addition, it states that it is able to play the roles \( R_4 \), \( R_7 \), and \( R_1 \) if required. The service described by Table 1 matches this query because the provider is able to play one of the roles \( (R_i) \) required by the requester and the requester can play some roles \( (R_a \) and \( R_i) \) that make true the necessary role conditions.

<table>
<thead>
<tr>
<th>Main Roles</th>
<th>( R_2 \lor (R_6 \land R_4) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capabilities</td>
<td>( R_6, R_7, R_1 )</td>
</tr>
</tbody>
</table>

More formally, we describe a registered service advertisement \( S \) and a service request \( R \) as:

\[
S = \text{SET OF InteractiveRoleDescription} \\
\text{InteractiveRoleDescription} = \langle \text{MainRole}, \text{RoleExpression} \rangle \\
\text{RoleExpression} = \text{SET OF ConjunctiveRoleList} \\
\text{ConjunctiveRoleList} = \text{SET OF Roles} \\
\]

\[
R = \langle \text{MainRoleR}, \text{Capabilities} \rangle \\
\text{MainRoleR} = \text{RoleExpression} \\
\text{Capabilities} = \text{SET OF Roles} \\
\]

Notice that our approach is compatible with services that do not make use of the role-based extensions in their description. In case a service description does not include the role-based approach, we assume it has a main role *Communicator* (the top
and most general concept of the ontology) and no necessary roles are required from the requester. If the request does not include a role description, we assume the requester is not interested in the role-based approach and the matchmaker will omit that phase in the service matching process.

3.3 Extending OWL-S with roles

We use the service description language OWL-S [16] in our system. An OWL-S service description consists of three parts: the service profile used for advertising and discovering services; the process model, which details how the service operates; and the service grounding, which provides details on how to interoperate with the service.

The service profile is designed to describe what the service does, so this is the adequate location for role descriptions. A service profile basically includes information about inputs, outputs, preconditions and effects, as well as non-functional information like service category, service classification, service product, etc. However, organisational information such as roles or interactions does not fit in any of those predefined fields, so we opt for including the role description as an additional parameter, called ServiceRoles in the case of service descriptions and QueryRoles for service requests.

Table 3 shows a role-based service advertisement for the second opinion example. Fig. 5 shows a part of the corresponding OWL-S service profile.

Table 3. Second Opinion role-based service advertisement

<table>
<thead>
<tr>
<th>Main Role</th>
<th>Advisor</th>
<th>Explainer</th>
<th>Informer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Necessary roles</td>
<td>Informer</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4 shows an example in which an agent is looking for a second opinion provider. The request specifies that the requester is able to play the informer and explainer roles if necessary.

Table 4. Second Opinion role-based service request

<table>
<thead>
<tr>
<th>Main Role</th>
<th>SecondOpinionRequestee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capabilities</td>
<td>Informer, Explainer</td>
</tr>
</tbody>
</table>
4. Role-based Service Matching Algorithm

We have developed a role-based matching algorithm that takes as inputs a service request \((R)\) and a service advertisement \((S)\), and returns the degree of match \((\text{dom})\) between them. Essentially, it searches the role in the advertisement \(S\) that best matches the one in the query \((R)\). The match between a role in the query and one role in the advertisement is made by the function \(\text{MatchAtomicRequest}\), which also receives the set of capabilities of the requester, in case it needs to check if the necessary roles can be provided by the requester.

Fig. 6 shows a pseudocode for the matching algorithm \((\text{Match} \text{ function})\). As described before, the request may not only include a role but also an expression (a disjunction of conjunction of roles). The loops in lines 4 and 6 decompose that expression, using the minimum as combination function for the values in a conjunction and the maximum for disjunctions.
Fig. 6. Matching algorithm. Returns the degree of match between a request $R$ and a service advertisement $S$

$MatchAtomicRequest$ returns the degree of match between a role in the request and a service advertisement, given the set of capabilities of the requester. It compares the requested role with every other given role and returns the maximum degree of match. For each role in the advertisement, the match between the main roles is made, as well as the match between the necessary roles and the capabilities of the requester. The minimum of both values is considered the degree of match. Again, the necessary roles are given by a logical expression, which must be evaluated by decomposing it within the $MatchRoleExpr$ function.
The semantic match of two roles $R_A$ (advertisement) and $R_Q$ (query) is made based on the ontology of roles. It is a function that depends on two factors:

1. Level of match. This is the (subsumption) relation between the two concepts $(R_A, R_Q)$ in the ontology. We differentiate among the four degrees of match proposed by Paolucci et al. [17]:
   - exact: if $R_A = R_Q$
   - plug-in: if $R_A$ subsumes $R_Q$
   - subsumes: if $R_Q$ subsumes $R_A$
   - fail: otherwise

2. The distance (number of arcs) between $R_A$ and $R_Q$ in the taxonomy.

We consider, for the moment, that all roles have the same importance and that the generality (depth in the taxonomy) of the roles is not relevant. We combine both criteria into a final degree of match which is a real number in the range $[0, 1]$, so service providers can be selected by simply comparing these numbers. In this combination, the level of match always has higher priority: the value representing the degree of match is equal to 1 in case of an exact match, it varies between 1 and 0.5 in case of a plug-in match, rests between 0.5 and 0 in case of a subsumes match, and it is equal to 0 in case of a fail.

There are infinite functions that fulfill that precondition. One equation that implements this behaviour is shown in Fig. 7, where $x$ is the distance between $R_A$ and $R_Q$ ($\text{depth}(R_A) - \text{depth}(R_Q)$) in the role ontology (if there is a subsumption relation between them). This kind of function guarantees that the value of a plug-in match is always greater than the value of a subsumes match, and it only considers the distance between the two concepts, rather than the total depth of the ontology tree\(^1\), which may change depending on the domain. Furthermore, the smaller the distance between concepts (either in the case of plug-in or subsumes match), the more influence will have a change of distance in the degree of match.

Consider, for instance, the example case described in section 3. The degree of match between the advertised role Advisor and the required role SecondOpinionRequestee (that are related by a subsumption relation like the interactions Advisement and SecondOpinion) is:

\[
x = \text{depth}(\text{Advisor}) - \text{depth}(\text{SecondOpinionRequestee}) = 4 - 2 = 2
\]

\[
\text{dom}(x) = \frac{1}{2} + \frac{1}{2 \cdot \exp(2)} = 0.5677
\]

The necessary role Informer is included in the capabilities set of the requester, so an exact match (1) is obtained. The final degree of match is the minimum of both values, i.e., 0.5677, which corresponds to a plug-in match.

\[^1\text{Note that, for instance, if a linear function is used, the maximum possible distance between two concepts must be known a priori to establish the equation (e.g. } \text{dom}(x) = 1 - x/6).\]
5. Conclusions and future lines

In this paper, we have described a new approach for service description and discovery in multiagent systems based on organisational models. We have identified roles and interaction ontologies as key information to this respect, have applied them to service advertisements and requests, and have formalised both on the basis of OWL-S. Furthermore, we have presented an extension to common service matching techniques so as to exploit this additional organisational information. We have provided examples to demonstrate that such an approach is relevant for real-world scenarios.

We are considering some variations in several aspects of the algorithm presented in order to improve its precision. In particular, we will study how the matchmaker behaves if other aggregation functions are used instead of minimum and maximum to combine individual roles degrees of match. Ideas from the fuzzy logic field (t-norms/t-conorms) will be considered during the validation.

Organisational concepts, such as roles and interactions \[12\], are abstraction mechanisms that have been used for many years in object oriented models, and are now usually present in agent-oriented design methodologies \[21\], \[6\], \[7\]. However, current service discovery approaches \[14\], \[17\], \[20\] do not make use of those abstractions to describe and search service providers. To the best of our knowledge there are no service discovery approaches that exploit role and interaction type information in multiagent systems.

We have constructed a matchmaker that implements the algorithm presented in this paper, paying special attention to its efficiency by maintaining a matrix of pre-calculated degrees of match between every pair of generic roles in the ontology. We are currently investigating the best way of combining our matchmaker and
OWLS-MX in terms of efficiency and effectiveness. At least two possibilities are being considered: (i) pre-filter configuration and (ii) result combination. For the latter case, different aggregation functions are being considered.

In the future, we plan to develop a design method for agent-based service description from an organisational perspective. We also plan to explore as to how far our proposal is compatible with the WSMO [22] service description approach.

Acknowledgments

This work has been partially supported in part by the European Commission under grant FP6-IST-511632 (CASCOM), and by the Spanish Ministry of Education and Science, project TIC2003-08763-C02-02

References

22. WSMO working group http://www.wsmo.org/.