# FORMAL METHOD FOR AUTOMATIC AND SEMANTIC MAPPING OF DISTRIBUTED SERVICE-ONTOLOGIES

Nacima Mellal and Richard Dapoiny
*LISTIC/ Polytech'Savoie, universit de Savoie, P.B. 80439, 74944, Annecy Cedex*

Keywords: Services, distributed ontologies, automatic and semantic mapping, Information Flow (IF) model.

Abstract: Many distributed heterogenous systems exchange information between them. Currently, most of them are described in terms of ontologies. When ontologies are distributed, arises the problem of achieving sematic interoperability. This is undertaken by a process which defines rules to relate these ontologies, called "Ontology Mapping" in order to achieve a given goal. This paper describes a methodology for automatic and semantic mapping of ontologies. Our main interest is focused on ontologies describing services of systems. These ontologies are called "Service Ontologies". So, we investigate an approach where the mapping of ontologies provides full semantic integration between distributed service ontologies using Information Flow model.

## 1 INTRODUCTION

In the Artificial Intelligence field, exchange of information between distributed systems is a challenging theme. Systems usually need to interoperate, they also need to understand what they exchange. This introduces the notion of Semantic Interoperability. To reach this last, information must be expressed in a formal way. Ontologies seem good mean to achieve this. An ontology is generally seen as an explicit specification of a conceptualization (Gruber, 1993), (Fikes, 1996). It should give an explicit definition of concepts and relations between them. In order to exchange information basing on the use of ontologies, this practice of finding correspondences between ontologies is called "Ontology Mapping". It arises in many application scenarios. In (Doan and Halevy, 2004) authors have focused on the semantic integration work in the database community. Where, in (Noy and Stuckenschmidt, 2005) this issue is studied by giving a brief review of ontology-based approaches to semantic integration. In (Kalfoglou and Schorlemmer, 2003), a formalization of the coordination process between ontologies is described, based on exchange that captures progressive partial semantic integration, using the IF model (Barwise and Seligman, 1997). Most of these works propose semi automatic mappings to reach the semantic interoperability. Thus, it is necessary to develop automatic techniques for mapping ontologies. Our approach shares the idea in (Kalfoglou and Schorlemmer, 2003), which uses of IF Model to solve semantics coordination of ontologies in distributed systems. We propose a methodology allowing automatic mapping between distributed ontologies. As a crucial topic, information exchange between ontologies must occur in a semantic and sound manner. The IF model is mainly based on a theory, called IF Theory. This latter introduces a consequence relation $\vdash$ on a set of types, so that we can retrieve from a type $t_1$ the corresponding type $t_2$ by $\vdash$, $t_1$ and $t_2$ belonging to different sets of types. Precisely, how the IF model selects two entities A and B, located respectively in two systems. To relate these entities, the model starts from the type of A, finds out what type is related to it in the second system . Then, finds the corresponding entity B thanks to the relation $\vdash$. Entities of two systems are linked by an information channel.
As many nowadays distributed systems are based on the notion of services, we describe systems by a set of functionalities offered by its components as services. Certainly, services may depend on others of the same system or of different systems. An efficient and promising way to implement this is through the use of ontologies. In our approach, concepts of ontologies are services, relationships between them express the functional dependencies among them. Our interest is to achieve the semantic interoperability when services depend to others in a distributed system using IF model to connect services in order to solve a high level service.
In this paper, we introduce in section 2 the notion of Service Ontology, illustrating it by an example. Then, we describe in section 3 the Ontology Mapping mechanism. We conclude by some perspectives.



Mellal N. and Dapoiny R. (2007).
FORMAL METHOD FOR AUTOMATIC AND SEMANTIC MAPPING OF DISTRIBUTED SERVICE-ONTOLOGIES.
In *Proceedings of the Second International Conference on Software and Data Technologies - PL/DPS/KE/WsMUSE*, pages 259-263
DOI: 10.5220/0001329602590263

# 2 SERVICE ONTOLOGY

## 2.1 Case Study

The example will illustrate the different definitions proposed in this paper. We consider a distributed system which employs ticket agents. Each agent is situated in a sub-system. Agents attempt to achieve some services for distributing, selling, buying, booking tickets from a range of sources. They may communicate and exchange services on the web. We will treat a case where an agent (*Agent*$_1$) attempts to achieve a service of buying a ticket to go from Annecy (France) to Barcelona (Spain). Once the depart and the destination are identified, agent may obtain the itinerary, but how to do if the agent can not achieve the service of obtaining the itinerary? We will give details about this problem in the next sections.

## 2.2 Service Notion

In (Kitamura et al., 2004), authors denoted a functionality of a component as a "verb+noun" style for representing the components activities. Following this approach, we associate with each service some possible actions in order to fulfill the intended service (Hertzberg and Thiebaux, 1994), (Lifschitz, 1993). So, we describe each service as a tuple: (Action verb, {Property, Object}), where the action verb acts on the object's property. For each object, corresponds a type defining its classification domain. We denote a type an "Object Type" and the object itself specified by an identity is denoted "Object Token". The tuple {Property, Object} is called a context, when the object is an object type, we call the context "Context Type", otherwise, "Context Token".

**Definition 1** *"Context Type (respectively, Context Token)"*
*A context type $\xi_i$ (resp, context token) is a tuple:* $\xi_i \stackrel{def}{\equiv} (p, \{\psi_1, \psi_2, ...\psi_n\})$*. Where p is a property,* $\psi_1$, $\psi_2$, ... $\psi_n$ *is a set of object types (resp, object tokens).*

Following the example, we suppose that the distributed system is represented by two ticket agents. Thus, we propose some context types to the agent *Agent*$_1$:
$\xi_1 = (Departure, Trip)$, $\xi_2 = (Destinaton, Trip)$, $\xi_3 = (Date, Trip)$, $\xi_4 = (Itinerary, Trip)$, and $\xi_5 = (Price, Trip)$.
Context tokens are not limited to:
$c_1 = (Annecy, Flight - Trip)$, $c_2 = (Barcelona, Flight - Trip)$, $c_3 = (July - 22 - 2007, Flight - Trip)$, $c_4 = (Itinerary_1, Flight - Trip)$, and $c_5 = (200euros, Flight - Trip)$.

**Remark**: We propose the same context types and tokens to *Agent*$_2$ replacing $\xi$ by $\xi'$ and $c$ by $c'$.
The problem of finding a formal structure to the notion of service is treated in several researches. In (Umeda et al., 1996), author suggested a representation of the form "To do X" for intended services. Following this principles, we define a service such as:

**Definition 2** *"Service Type (resp, Service Token)"*
*A Service type (resp, Service token), denoted by $\gamma_i$, is defined as a pair:*
$\gamma_i \stackrel{def}{\equiv} (\mathsf{a}, \{\xi_1, ..., \xi_k\})$*. Where* a*, is an action verb,* $\Xi$*, a non-empty set of context types (resp, context tokens) and* $\{\xi_1, ..., \xi_k\} \subseteq \Xi$.

According to our example, there are basically some primitive service types. Some are given to *Agent*$_1$:
$\gamma_1 = (\mathsf{to\_identify}, \{\xi_1\})$, $\gamma_2 = (\mathsf{to\_identify}, \{\xi_2\})$,
$\gamma_3 = (\mathsf{to\_identify}, \{\xi_3\})$, $\gamma_4 = (\mathsf{to\_obtain}, \{\xi_4\})$
and $\gamma_5 = (\mathsf{to\_obtain}, \{\xi_5\})$
Some service tokens for *Agent*$_1$:
$\theta_1 = (\mathsf{to\_identify}, \{\mathsf{c}_1\}$, $\theta_2 = (\mathsf{to\_identify}, \{\mathsf{c}_2\}$,
$\theta_3 = (\mathsf{to\_identify}, \{\mathsf{c}_3\}$, $\theta_4 = (\mathsf{to\_obtain}, \{\mathsf{c}_4\}$
and $\theta_5 = (\mathsf{to\_obtain}, \{\mathsf{c}_5\}$
**Remark**: As a special case, we propose the same service types and tokens to *Agent*$_2$ replacing $\gamma$ by $\gamma'$ and $\theta$ by $\theta'$.

## 2.3 Service Ontology

An ontology is a description of the concepts and relationships between them. In our context, ontology is described by:

1. ***Concepts***: We associate service types with ontology concepts.
2. ***Relations***: An ontology is related to a Gentzen system, which is a deduction system expressed by the first order logic. The notion of *sequent* is central in Gentzen system. Given a set *S*, a sequent of *S* is a pair $\langle X, Y \rangle$ of subsets of *S*. A binary relation $\vdash$ between subsets of *S* is called a consequence relation on *S*. The syntax of sequent is $X \vdash Y$ [1]. Sequents will represent subsets of ontology concepts (service types), $\vdash$ will express the functional dependency between these services.

   **Definition 3** *"Functional Relation $\vdash$"*
   *Let C be a non empty set of ontology concepts. The binary relation $\vdash$ between subsets of C is the Gentzen consequence relation on C, such that:*

[1] The notation $X \vdash Y$ may be written by $x_i, x_j, .. \vdash y_a, y_b, ..$, where $x_i, x_j, .. \in X$ and $x_i, x_j, .. \in Y$. The comma on the left is interpreted like a conjunction, the comma on the right like a disjunction

*Two subsets of $C$: $c_x$ and $c_y$ are related by $\vdash$, denoted $c_x \vdash c_y$, iff the service types of $c_x$ influence functionally service types belonging to $c_y$.*

The expression $c_x \vdash c_y$ must be understood that the only way to achieve services in $c_y$ is to have already achieved services in $c_x$.

Therefore, we propose the following definition of a service ontology:

**Definition 4** *"Service Ontology"*
*A Service Ontology "SO" is a tuple $< C, R >$, where $C$ is the set of ontology concepts and $R$ is the set of relations. SO is described by an oriented graph, where, its nodes represent concepts, and edges linking nodes represent $\vdash$.*

The defined *SO* describes a set of concepts, service types, and the relations between them. This set can be seen as complex service which is called *global service* and its elements (ontology concepts) as sub-services. We deduce that a global service is associated with a *SO*. It is a particular service token. We detail two global services $\Theta_1$ related to $Agent_1$ and $\Theta'_1$ for $Agent_2$. $\Theta_1 = (to\_buy, \{ticket, Flight-Trip\})$ represented by the service ontology $SO_1$. $\Theta'_1 = (to\_obtain, \{Duration, Flight-Trip\})$ is represented by $SO_2$, (see figure 1). In this section, we described

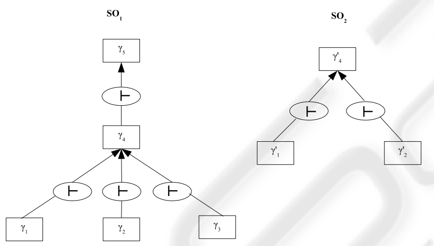

Figure 1: Service Ontologies of $\Theta_1$ and $\Theta'_1$.

that the aim of an ontology of services is to describe systems. The question which should be asked is how to map between ontologies if they describe distributed systems?

# 3 ONTOLOGY MAPPING

The approach is presented by different steps: Formal description of services, building service-ontologies, building the information channel to the given application, and identification of the logic on the core of the information channel and its distribution on this channel. Before developping the first and second steps, we invite the reader to see (N. Mellal, 2006) and (Barwise and Seligman, 1997) for more details on IF Model.

Table 1: The binary relation $\models$ of IF-$C_1$ and IF-$C'_1$.

| | $C'_1$ | | | | $C'_1$ | | |
|---|---|---|---|---|---|---|---|
| | $\gamma'_1$ | $\gamma'_2$ | $\gamma'_4$ | | $\gamma'_1$ | $\gamma'_2$ | $\gamma'_4$ |
| $c'_1$ | 1 | 0 | 0 | $c'_1$ | 1 | 0 | 0 |
| $c'_2$ | 0 | 1 | 0 | $c'_2$ | 0 | 1 | 0 |
| $c'_3$ | 0 | 0 | 0 | $c'_3$ | 0 | 0 | 0 |
| $c'_4$ | 0 | 0 | 1 | $c'_4$ | 0 | 0 | 1 |
| $c'_5$ | 0 | 0 | 0 | $c'_5$ | 0 | 0 | 0 |

## 3.1 Building the Information Channel

### 3.1.1 Identification of IF Classifications

For each global service described by a service ontology, we associate an IF classification. For the agent $Agent_1$, we associate the classification $C_1$ to $\Theta_1$ and $C'_1$ to $\Theta'_1$, see table 1:

### 3.1.2 Identification of IF Theories

IF Theories describe the different constraints on information flowing in the system. In our context a constraint on service types is denoted $\gamma_i \vdash \gamma_k$ represents the fact that $\gamma_k$ depends functionally on $\gamma_i$. According to our objectives, IF theory specifies service ontology. We give in the following the IF theories of agents $Agent_1$ and $Agent_2$: For ($Agent_1$), we have $\gamma_1 \vdash \gamma_4, \gamma_2 \vdash \gamma_4, \gamma_3 \vdash \gamma_4$ and $\gamma_1, \gamma_2, \gamma_3 \vdash \gamma_4, \gamma_4 \vdash \gamma_5$
For ($Agent_2$), we have $\gamma'_1 \vdash \gamma'_4, \gamma'_2 \vdash \gamma'_4, \gamma'_1, \gamma'_2 \vdash \gamma'_4$

### 3.1.3 Construction of IF Channel

It is the central aspect in the process of mapping between service ontologies. In this step, we aim to achieve $\Theta_1$. As assumed in section (2), $Agent_1$ cannot obtain the itinerary from Annecy to Barcelona, so the service token $\theta_4 = (to\_obtain, \{c_4\})$ is not achievable locally. Therefore, we will assume a partial alignment of context tokens. $c_4$ will be connected with the context tokens candidates where their types are related by the $\vdash$, so having the same type as $c_4$. These candidates ($c'_j, ..$) are obtained from remote systems. Let us code that we may find service types having the same context type as the context type of $c_4$ or not. It is formalized by the classification $A$ given in the table 2. Type of $A$ is $c_4$ and its possible tokens are $a$ and $b$. $A$ plays the role of reference to compare types of distributed classifications. In our case, we compare types of $C_1$ with those of $C'_1$. Let us note that in this application, we get only two cases, but in general for $m$ context tokens we will get $2^m$ different cases. Our aim is to relate via infomorphisms the context token $c_4$ appearing as a token in $C_1$ with those of

Table 2: $A$ and its flip $A^{\perp}$.

$$A: \begin{array}{c|c} \models_A & c_4 \\ \hline a & 0 \\ b & 1 \end{array} \qquad A^{\perp}: \begin{array}{c|cc} \models_{A^{\perp}} & a & b \\ \hline c_4 & 0 & 1 \end{array}$$

distributed classifications ($C_1'$). We note that $c_4$ is appeared as a type in $A$, where the context tokens in the other classifications are classified as tokens and not as types. That is, it is useful to introduce the flip of the classification $A$, by interchanging rows and columns. See the table 2. Thus, this gives rise to the respective infomorphisms $\zeta_1^{(Agent_1)}, \zeta_1'^{(Agent_2)}$ permit to connect $A^{\perp}$ with $C_1$ and $C_1'$. $\zeta_1^{(Agent_1)} : A^{\perp} \rightarrow C_1^{(Agent_1)}$ and $\zeta_1'^{(Agent_2)} : A^{\perp} \rightarrow C_1'^{(Agent_2)}$
Applying these infomorphisms we find:
**with** $C_1$: $\zeta_1^{\wedge}(\gamma_1) = a$, $\zeta_1^{\wedge}(\gamma_2)=a$, $\zeta_1^{\wedge}(\gamma_3)=a$, $\zeta_1^{\wedge}(\gamma_4)=b$, $\zeta_1^{\vee}(c_4)= c_4$,
**with** $C_1'$: $\zeta_1'^{\wedge}(\gamma_1') = a$, $\zeta_1'^{\wedge}(\gamma_2') = a$, $\zeta_1'^{\wedge}(\gamma_4') = b$, $\zeta_1'^{\vee}(c_4) = c_4'$.
The alignment allows the generation of the desired channel between $C_1^{(Agent_1)}$ and $C_1'^{(Agent_2)}$. A core classification C is built with a couple of infomorphisms:

$g_1^{(Agent_1)} : C_1^{(Agent_1)} \rightleftarrows C$ *and* $g_2^{(Agent_2)} : C_1'^{(Agent_2)} \rightleftarrows C$. Types of C are the elements of the disjoint union of types from $C_1^{(Agent_1)}$ and $C_1'^{(Agent_2)}$. Tokens are the cartesian product of tokens in $C_1$ and tokens in $C_1'$. According to our scenario, $c_4$ of type $\gamma_4$ is connected to a token $c_q$ of type $\gamma_x$ in the classification $C_1'^{(Agent_2)}$ to form the pair $(c_4, c_q)$ iff $\zeta_1^{\wedge}(\gamma_4)=\zeta_1'^{\wedge}(\gamma_x')$. This condition means that $\gamma_4$ and $\gamma_x'$ are of the same type in the classification $A^{\perp}$. As a result, we have the pair: $(c_4, c_4')$. The IF theory on the core is built on the union of types. The theory expresses how the types of $C_1$ are related logically to the types of $C_1'$ on the core of the information channel. The IF theory relates $\gamma_4$ with the types in $C_1'$. As a result, we have one constraint: $\gamma_4' \vdash \gamma_4$ for the sum $C_1, C_1'$. In this example, we find only one constraint, but it is possible to find more than one and agent will choose the correspondent service type, to achieve the global service.

## 3.2 Identification of the Logic on the Core and the Distributed IF Logic

Given the logic Log(C)=L on the core C, the distributed logic $DLog(C)$ on the sum of classifications $C_1^{(Agent_1)}+C_1'^{(Agent_2)}$ is the inverse image of $Log(C)$ on this sum. In other words, the inverse image of the IF logic in $C$ is the result of the co-product of $C_1^{(Agent_1)}$ and $C_1'^{(Agent_2)}$ with the morphism $[g_1^{(Agent_1)}, g_2^{(Agent_2)}]^{-1}$. We obtain sequents like $(\gamma_q'^{(Agent_2)}, \gamma_4^{(Agent_1)})$ relating service types on remote systems to the local service type $\gamma_4$. This result describes the semantic interoperability. According to the initial constraint on service tokens the sequent $(\gamma_4'^{(Agent_2)}, \gamma_4^{(Agent_1)})$ matches the conditions. Therefore, the $C_1$ has to be mapped semantically to $C_1'$ in order to constitute a sound distributed service.

# 4 CONCLUSION

We have presented in the present paper a formal method for mapping distributed service ontologies in a sound and automatic manner, basing on IF model. In (N. Mellal, 2006), we proposed an algorithm specifying the process of mapping among multi agent systems which represent distributed systems. Each agent is situated in a a system and has its own service ontologies. In this work, the IF-based approach tackles the problem of building these dependencies from distributed logics. Future work includes implementing a multi agent system to achieve automatically the semantic mapping of service ontologies.

# REFERENCES


Barwise, J. and Seligman, J. (1997). *Information Flow: The Logic of Distributed Systems*. Cambridge University Press.

Doan, A. and Halevy, A. (2004). Semantic-integration research in the database community. *Special issue on semantic integration*, 26:83–94.

Fikes, R. (1996). Ontologies: What are they, and where's the research?

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220.

Hertzberg, J. and Thiebaux, S. (1994). Turning an action formalism into a planner: a case study. *Journal of Logic and Computation*, pages 617–654.

Kalfoglou, Y. and Schorlemmer, M. (2003). If-map: an ontology mapping method based on information flow theory. *Journal on Data Semantics*, 1(1):98–127.

Kitamura, Y., Kashiwase, M., Fuse, M., and Mizoguchi, R. (2004). Deployment of an ontological framework of functional design knowledge. *Advanced Engineering Informatics*, 18(2):115–127.

Lifschitz, V. (1993). A theory of actions. In *the tenth International Joint Conference on Artificial Intelligence*, pages 432–437.

N. Mellal, R. D. (2006). A multi-agent specification for the goal-ontology mapping in distributed complex systems. In *7th International Baltic Conference on Databases and Information Systems*, pages 235–243.

Noy, N. and Stuckenschmidt, H. (2005). Ontology alignment: An annotated bibliography.

Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y., and Tomiyama, T. (1996). Supporting conceptual design based on the function-behavior-state modeler. pages 275–288.