An Application of Approximate Ontology Matching in eResponse

Maurizio Marchese
University of Trento
maurizio.marchese@disi.unitn.it

Lorenzino Vaccari
University of Trento
vaccari@disi.unitn.it

Pavel Shvaiko
University of Trento
pavel@disi.unitn.it

Juan Pane
University of Trento
pane@disi.unitn.it

ABSTRACT

Ontology matching is a key problem in many metadata intensive application domains, including emergency response, data integration, peer-to-peer information sharing, web service composition, and query answering on the web. In this paper we present an emergency response scenario based on the organizational model as used in Trentino region, Italy. We provide a formalization of this scenario with the help of lightweight coordination calculus. Then, we discuss an automatic approximate structure preserving matching algorithm which we applied within the emergency response scenario. The evaluation results, though preliminary, are encouraging.

Keywords

Peer-to-peer networks, web services, semantic heterogeneity, ontology matching, knowledge sharing in crisis management, interaction modeling.

INTRODUCTION

The need to harness the potential of electronic networks in emergency situations is widely recognized as a relevant research priority. *In times of crisis - be it a natural disaster, terrorist attack or infrastructure failure - mobile personnel need to work together in time-critical and dangerous situations. Real-time access to information and knowledge will help save lives. Crises are complex situations, with large numbers and varieties of mobile personnel - medical and rescue teams, police, fire fighters and other security personnel - appearing on the spot at short notice. These different teams come from different organizations, and generally have incomplete or even contradictory knowledge of the crisis situation*[1]. The quoted text provides some examples of the key elements needed in emergency response situations. In particular, emergency management activities – that in the following we will reference as emergency response (eResponse) activities – are developed and implemented through the essential analysis of information and the coordination of the involved peers, including emergency personnel, army, volunteers, etc. Disaster data and events can be acquired, modelled, fused and displayed in state-of-the-art Spatial Data Infrastructures (SDIs) using distributed data sources, the majority of which are spatial. Moreover, emergency monitoring and management activities normally involve a range of different organizations and teams at various administrative levels with their own systems and services. The application of numerous and different actors, policies, procedures, data standards and systems, results in coordination problems with respect to data analysis, information delivery and resource management, all critical elements of emergency response management.

Current technologies and information systems can provide parts of the solution with ad-hoc and mostly centralized systems. In particular, institutional agencies (e.g., municipal, regional) have started to adopt distributed SDIs (Groot and McLaughlin, 2000, Nebert, 2004, Bernard et al., 2005). While Geographical Information Systems (GIS) are self-contained systems in which data and software applications are used mainly internally, the goal of SDI is to support the *interoperability* among different kinds of institutions, users and roles. However, SDIs are pervaded by *interoperability problems,* including: *(i)* geo-data interoperability, specifically, geographical datasets have particular properties (e.g., maps as implicit interfaces) to be tackled, which are different from other types of data (Parent et al., 2006) and *(ii)* geo-service interoperability issues, such as geo-service discovery, integration and composition.

[1] EU "Emergency Response Grid" programme: http://cordis.europa.eu/ist/grids/emergencey_response_grid.htm, 2006.

In this paper we propose to use a peer-to-peer (P2P) infrastructure for the eResponse domain. At the core of our approach is a specific view on semantics of both web service and agent coordination as proposed in (Robertson et al., 2007). Peers share explicit knowledge of the "interactions" in which they are engaged and these models of interaction are used operationally as the anchor for describing the semantics of the interaction. Instead of requiring a universal semantics across peers we require only that semantics is consistent (separately) for each instance of an interaction. These models of interactions are developed locally by peers and are shared on the network. Then, since there is no a priori semantic agreement (other than the interaction model), matching is needed to automatically make semantic commitments between the interacting parts. In particular, it is used to identify peers, which are suitable to play a particular role in an interaction model. For example, let us consider *i-th* interaction model IM_i, where a constraint on playing *m-th* role r_m in IM_i is as follows: *T1:* `getMap(MapFile, Version, Layers)`. Let us suppose that *T2:* `getMap(Dimension(Width, Height), MapFile, Edition, Layers)` is a description of the capabilities of *k-th* peer, p_k. Then, p_k wants to subscribe to r_m in IM_i, and thus, its capabilities should be matched to the constraints of r_m. If the matching score between *T1* and *T2* in the [0 1] range (e.g., 0.58) is good enough, for instance, higher than an empirically established threshold, such as 0.5, then, peer p_k can be allowed to play role r_m.

The main contributions of the paper include: *(i)* a P2P formalization of the eResponse scenario based on the organizational model as used in the Trentino region, Italy, *(ii)* an application of the approximate semantic matching algorithm, originally proposed in (Giunchiglia et al., 2007b), within the eResponse settings.

The structure of the rest of the paper is as follows. We introduce our emergency response scenario and its formalization in Section 2 and Section 3, respectively. In Section 4 we present the use of an approximate structure preserving semantic matching algorithm in order to reduce the semantic heterogeneity problem as required by the scenario. Preliminary evaluation of the approach used is provided in Section 5. The related work is discussed in Section 6. Finally, the major findings of the paper are summarized in Section 7.

SCENARIO

We have analyzed the organizational model of the distributed GIS Agency infrastructure of Trentino. The framework of the distributed system is represented by a number of specialized GIS agencies: civilian protection, urban planning, forestry, viability, etc. Each GIS agency is responsible for providing a subset of the geographic information for the local region. To support interoperability among the different GIS agencies the regional information infrastructure is shifting from a traditional GIS system to a SDI.

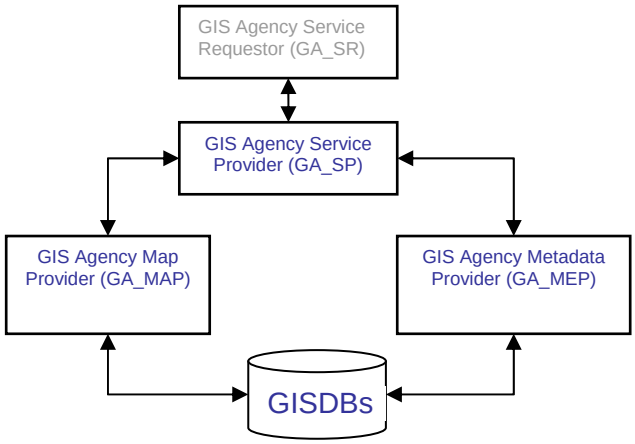

Figure 1. The Trentino GIS Agency organizational model.

We focus on the details and roles of the GIS Agency (GA). The GIS Agency is responsible to provide geographic datasets and services to external service requestors. The main generic actors in the current organization model of the GA Agency in Trentino, together with a short description of their main roles, are shown in Figure 2 and the following:

- GA_SR: (GIS Agency Service Requestor). Main role: ask for service (maps, datasets, analysis, etc.).
- GA_SP: (GIS Agency Service Provider). Main role: interface from external actors and internal GA actors. Service aggregation (design time / run time).

- GA_MAP: (GIS Agency Map Provider). Main role: building geographical maps from a number of available sources of geo-data (GeoDBs).
- GA_MEP: (GIS Agency Metadata Provider). Main role: GIS metadata provider (formalized description, search, matching, etc.) from a number of available sources of geo-data (GeoDBs).

Within the general Trentino SDI management scenario, we focus on the most commonly used specific use case, i.e., Map Request Service. Let us discuss one particular – but the most typical – request that can be made by the service requestor agent (GA_SR in Figure 1): a digital map request. A service requestor – both in an emergency or normal situation – needs to visualize a map of a region with geo-referenced information selected by a user. Therefore, the searched map is a composition of different geographic layers offered by one of the service provider agents (GA_SP in Figure 1).

A simplified interaction for the generic Map Request Service is illustrated in Figure 2. It uses a standard OGC (Open Geospatial Consortium) WMS (Web Map Service)[2] request and models a subset of the role interactions from an external requestor to the service provider for the request of a map service:

- The requestor assumes the GA_SR role and it asks the service provider (GA_SP) for the list and the characteristics of the services (Capabilities) provided by the service provider.
- The service provider provides its capabilities, in particular: the list of available services (AvailableServices), the list of geographic datasets managed by the server (AvailableLayers), the file format of the return services (Format), and the geographic limits of the available services (XMin_ME,YMin_ME,XMax_ME,YMax_ME).
- Then, the GA_SR asks for the map service using the information received from the previous step. The RequestMap message contains the URL of the requested map (MapFile), the version of the service (Version), the requested geographic layers (Layers), the dimension of the map (Width, Heigth), the graphic format of the map (Format), the spatial coverage of the map (XMin_BB, YMin_BB, Xmax_BB, YMax_BB).
- The service provider provides the map (Map) requested by the requestor.
- Finally, the service requestor asks for the graphic legend that describes the previous map and the service provider sends the legend (Legend) to the service requestor.

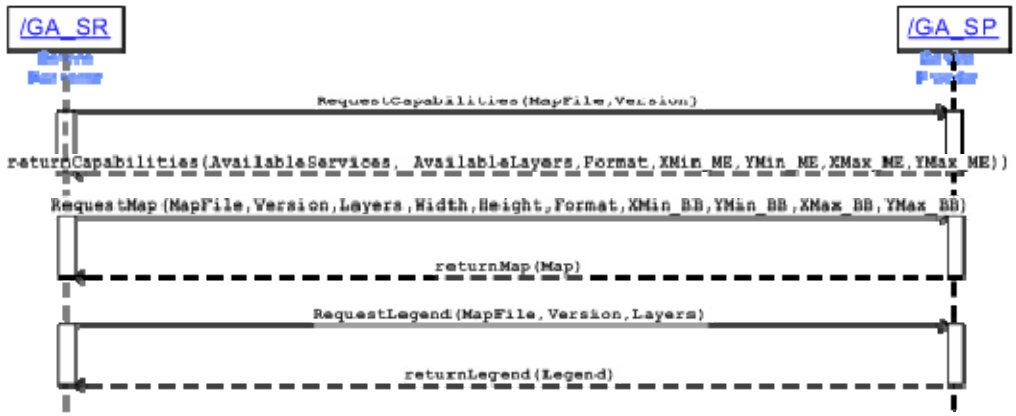

Figure 2. Sequence diagram for the map request service.

[2] Open Geospatial Consortium web site http://www.opengeospatial.org

INTERACTION MODELS

In this section, we first describe the Lightweight Coordination Calculus (LCC), that is, the communication language employed to implement the interactions among peers acting in our emergency network. We then focus on the selected composition problem depicted in the previous section and provide its formalization with LCC.

LCC is a protocol language used to describe interactions among distributed processes, e.g., agents, web services. LCC was designed specifically for expressing P2P style interactions within multi-agent systems, i.e., without any central control; therefore, it is well suited for modeling coordination of software components running in an open environment. Its main characteristics are flexibility, modularity and the neutrality to the distributed communication infrastructure (Robertson, 2004). The applicability of P2P style protocol modeling, such as LCC, in the eResponse domain is driven by its potential to deal with a knowledge-rich and dynamic environment. In fact, the nature of the eResponse domain is such that process-aware systems are beneficial to prevent chaotic and uncontrolled conditions. Nevertheless, taking into account adaptability is fundamental to handle unexpected situations (sudden road blockage, fast and unpredicted events, etc.) which most likely happen in emergency situations. The general vision of interaction protocols accounts for the "structured coordination" requirement of the problem. On the other hand, the adoption of models specifically designed to explicit interactions in a P2P fashion and passed through an underling open infrastructure accounts for the support to flexibility and dynamicity.

Interactions in LCC are expressed as message passing behaviors associated with roles. The most basic behaviors are to send or receive messages, where sending a message may be conditional on satisfying a constraint (pre-condition) and receiving a message may imply constraints (post-condition) on the agent accepting it. A basic LCC interaction is shown in Figure 3.

$$
\begin{aligned}
& a(r1, A1) :: \\
& \qquad ask(X) \Rightarrow a(r2, A2) \leftarrow need(X)\ then \\
& \qquad update(X) \leftarrow return(X) \Leftarrow a(r2, A2) \\
& a(r2, A2) :: \\
& \qquad ask(X) \Leftarrow a(r1, A1) \\
& \qquad return(X) \Rightarrow a(r1, A1) \leftarrow get(X)
\end{aligned}
$$

Figure 3. LCC fragment. Double arrows (⇐,⇒) indicate message passing, single arrows (←,→) indicate constraint satisfaction.

In Figure 3, the agent `A1` playing the role `r1` verifies if it needs the info `X` (pre-condition `need(X)`); if yes, `A1` - playing the role `r2` - asks the agent `A2` for `X` by sending the message `ask(X)`. `A2` receives the message, `ask(X)` from `A1` and then obtain the info `X` (pre-condition `get(X)`) before sending back a reply to `A1` through the message `return(X)`. After having received the message `return(X)`, `A1` updates its knowledge (post-condition `update(X)`).

The constraints embedded into the protocol express its semantics and could be written as first-order logic predicates (e.g., in Prolog) as well as methods in an object-oriented language (e.g., in Java). The characteristic of modularity allows separating the protocol from the agent engineering. While performing the protocol, peers can therefore exchange messages, satisfy constraints before (after) messages are sent (received) and jump from one role to another so that a flexible interaction mechanism is enabled still following a structured policy, which is absolutely necessary for team-execution of coordinated tasks.

Due to lack of space we discuss here only the LCC code related to `GA_SP` role (used in Figure 2), see Figure 4. Specifically, the GIS agency service provider `P` assumes the role `ga_sp` and waits for one of the following requests: `requestCapabilities`, `requestMap` or `requestLegend`. In the first case, after receiving the request from the service requestor, it builds its capabilities (`getCapabilities(MapFile, Version, AvailableServices, AvailableLayers, Format, Xmin_ME, YMin_ME, Xmax_Me, YMax_ME)`) and passes them to the requestor. In the second request the service provider builds a digital map (`getMap` constraint) and passes it to the service requestor. In the third case the service provider agent provides a legend for a collection of requested layers.

In the following section we will use the `getMap` constraint (highlighted in a box in Figure 4 to facilitate the presentation) as part of the motivating example to the structure preserving semantic matching approach.

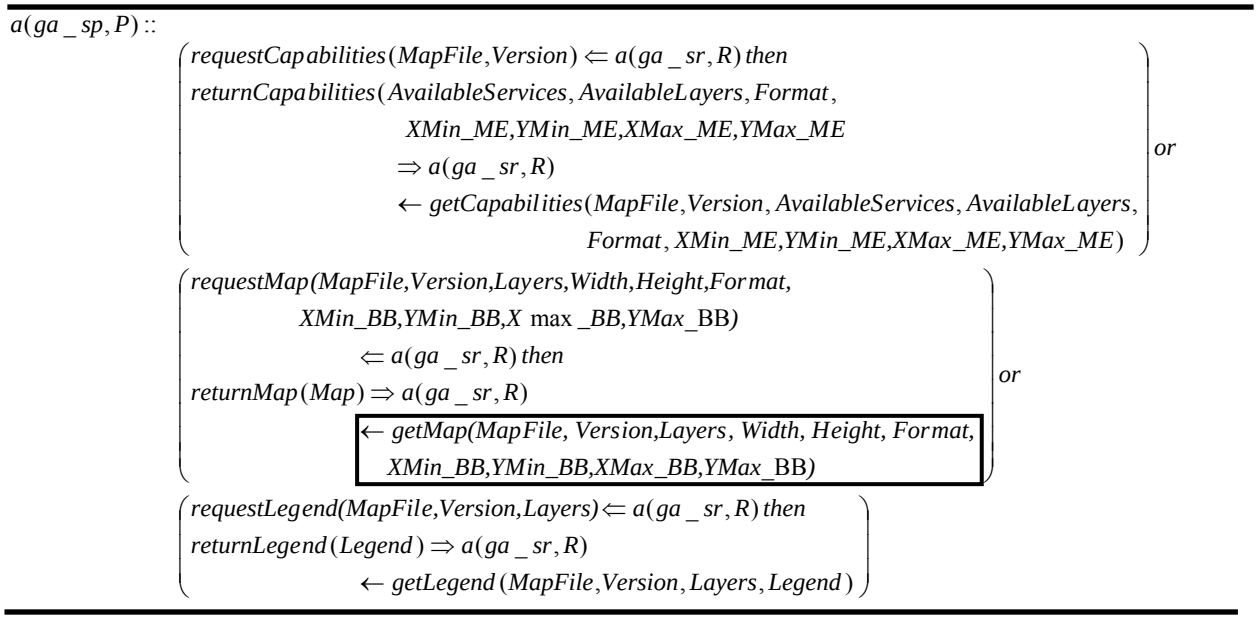

Figure 4. LCC fragment for the GIS agency service provider role.

APPROXIMATE STRUCTURE PRESERVING SEMANTIC MATCHING

An ontology typically provides a vocabulary that describes a domain of interest and a specification of the meaning of terms used in the vocabulary. Depending on the precision of this specification, the notion of ontology includes various data and conceptual models (Euzenat and Shvaiko, 2007). The term ontology is used here in a wide sense, and, hence, encompasses sets of terms, classifications, database schemas, thesauri, etc.

Ontology matching is a plausible solution to the semantic heterogeneity problem faced by information management systems. We view ontologies as graph-like structures (Giunchiglia and Shvaiko, 2003). We think of matching as an operation that takes two graph-like structures, such as web service descriptions, and produces a set of correspondences between the nodes of the graphs that correspond semantically to each other (Giunchiglia *et al.*, 2007a). Then, these correspondences can be used for various tasks, including data translation, etc. Thus, matching ontologies enables the knowledge and data expressed in the matched ontologies to interoperate.

In our scenario, the purpose of ontology matching is to reduce semantic heterogeneity in peer role descriptions. This scenario poses additional constraints on conventional ontology matching. Specifically, we need to compute the correspondences holding among the full graph structures and preserve certain structural properties of the graphs under consideration. Thus, the goal here is to have a *structure preserving semantic matching* operation. This operation takes two graph-like structures and produces a set of correspondences between those nodes of the graphs that correspond semantically to one another, (i) still preserving a set of structural properties of the graphs being matched, namely that functions are matched to functions and variables to variables; and (ii) only in the case if the graphs are globally similar to one another, e.g., $graph_1$ is *0.65* similar to $graph_2$ according to some measure.

Let us suppose that we want to match a constraint on a role, such as *T1*: `getMap(MapFile, Version, Layers, Width, Height, Format, XMin_BB, YMin_BB, XMax_BB, YMax_BB)` (see also box in Figure 4) with the capabilities of a peer willing to play that role, such as *T2*: `getMap(Dimension(Width, Height), MapFile, Edition, Layers, DataFormat, Request, Xmin, Ymin, Xmax, Ymax)`. These can be also viewed as web service descriptions. As shown in Figure 5, the first web service description requires the first argument of `getMap` function (`MapFile`) to be matched to the third (`MapFile`) of `getMap` function in the second description. `Version` in the first web service description must be passed to the second web service as the `edition` argument. Moreover, `request` on the right has no corresponding term on the left.

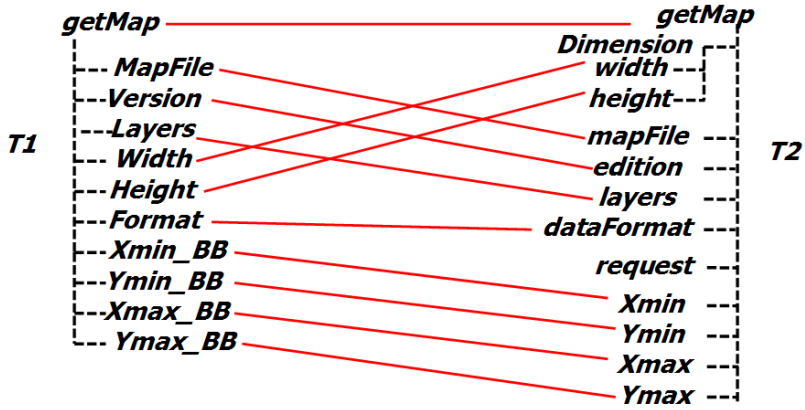

Figure 5. Two web service descriptions (trees) and correspondences (lines) between them.

The approach

The approach outlined next follows the work in (Giunchiglia *et al.*, 2007b). We briefly report it here for completeness and discuss it with the help of examples from the eResponse domain.

We focus on tree-like-structures, see, e.g., Figure 5. The matching process is organized in two steps: (i) node matching and (ii) tree matching. Node matching tackles the semantic heterogeneity problem by considering only labels at nodes and domain specific contextual information of the trees. We use here the S-Match system (Giunchiglia et al., 2007a). Technically, two nodes n1 and n2 in trees T1 and T2 match if and only if: c@n1 R c@n2 holds based on S-Match. c@n1 and c@n2 are the concepts at nodes n1 and n2. R $\in$ {=, $\sqsubseteq$, $\sqsupseteq$}. In particular, in semantic matching (Giunchiglia and Shvaiko, 2003) as implemented in the S-Match system (Giunchiglia et al., 2007a) the key idea is that the relations (e.g., =, $\sqsubseteq$) between nodes are determined by (i) expressing the entities of the ontologies as logical formulae and (ii) reducing the matching problem to a logical validity problem. Specifically, the entities are translated into logical formulas which explicitly express the concept descriptions as encoded in the ontology structure and in external resources, such as WordNet (Miller, 1995). This allows for a translation of the matching problem into a logical validity problem, which can then be efficiently resolved using sound and complete state-of-the-art satisfiability (SAT) solvers (Giunchiglia et al., 2005). Notice that the result of this stage is the set of correspondences holding between the nodes of the trees. For example, that getMap and Version in T1 correspond to getMap and edition in T2, respectively.
Tree matching, in turn, exploits the results of the node matching and the structure of the trees to find if these globally match each other. For obvious reasons we are interested in approximate tree matching. Technically, two trees *T1* and *T2* approximately match if and only if there is at least one node $n1_i$ in *T1* and node $n2_j$ in *T2* such that: (*i*) $n1_i$ approximately matches $n2_j$, (*ii*) all ancestors of $n1_i$ are approximately matched to the ancestors of $n2_j$.

The implementation

The implementation of approximate structure preserving matching is based on (*i*) a formal theory of abstraction and (*ii*) a tree edit distance. Specifically, the work in (Giunchiglia and Walsh, 1992) categorizes the various kinds of abstraction operations, including:

- *Predicate (Pd)*: Two or more predicates are merged, typically to the least general generalization in the predicate type hierarchy, e.g., *Height(X)* + *Dimension(X)* $\rightarrow$ *Dimension (X)*. We call *Dimension(X)* a predicate abstraction of *Height(X)*, namely *Dimension(X)* $\sqsupseteq_{Pd}$ *Height(X)*. Conversely, we call *Height(X)* a predicate refinement of *Dimension(X)*, namely *Height(X)* $\sqsubseteq_{Pd}$ *Dimension(X)*.
- *Domain (D)*: Two or more terms are merged, typically by moving constants to the least general generalization in the domain type hierarchy, e.g., *Xmin_BB* + *Xmin* $\rightarrow$ *Xmin*. We call *Xmin* a domain abstraction of *Xmin_BB*, namely *Xmin* $\sqsupseteq_D$ *Xmin_BB*. Conversely, we call *Xmin_BB* a domain refinement of *Xmin*, namely *Xmin_BB* $\sqsubseteq_D$ *Xmin*.

– *Propositional (P)*: One or more arguments are dropped, e.g., *Layers (L1)* $\rightarrow$ *Layers*. We call *Layers* a propositional abstraction of *Layers(L1)*, namely *Layers* $\sqsupseteq_P$ *Layers(L1)*. Conversely, *Layers(L1)* is a propositional refinement of *Layers*, namely *Layers(L1)* $\sqsubseteq_P$ *Layers*.

Let use consider the following example: (*Height(H)*) and (*Dimension*). In this case there is no abstraction/refinement operation that makes those first order terms equivalent. However, consequent applications of propositional and predicate abstraction operations make the two terms equivalent: *Height (X)* $\sqsubseteq_P$ *Height* $\sqsubseteq_{Pd}$ *Dimension*.

The abstraction/refinement operations discussed above allow us to preserve the desired properties: that functions are matched to functions and variables to variables. For example, predicate and domain abstraction/refinement operations do not convert a function into a variable. Thus, for instance, the correspondences between `Height` and `Width` in *T1* and `Dimension` in *T2*, although returned by the node matching, should be further discarded, and therefore, are not shown in Figure 5.

Then, the key idea is to use abstractions/refinements as allowed tree edit distance operations in order to estimate the similarity of two tree structures. Tree edit distance is the minimum number of tree edit operations, namely node *insertion*, *deletion*, *replacement*, required to transform one tree to another (Valiente, 2002). We want to: (*i*) minimize the editing cost, i.e., computation of the minimal cost composition of abstractions/refinements, (*ii*) allow only those tree edit operations that have their abstraction theoretic counterparts (in order to reflect semantics of first order terms).

A uniform proposal here is to assign the same unit cost to all operations that have their abstraction theoretic counterparts, while operations not allowed by definition of abstractions/refinements are assigned an infinite cost, see Table 1. The following three relations between trees are considered: *T1=T2, T1* $\sqsubseteq$ *T2*, and *T1* $\sqsupseteq$ *T2*. A global similarity (*TreeSim*) between two trees T1 and T2 is computed as follows:

$$TreeSim(T1,T2) = 1 - \frac{\min \sum_{i \in S} n_i \cdot Cost_i}{\max(size\,of\,T1, size\,of\,T2)}$$

where, *S* is the set of allowed tree edit operations, n_i is the number of *i*-th operations necessary to convert one tree into the other, $Cost_i$ is the cost of the *i*-th operation. The minimal edit distance is normalized by the size of the biggest tree. Finally, a normalized distance (denoting dissimilarity) is converted into a similarity score. For the case when *Cost* is infinite (see Table 1), *TreeSim* is estimated as zero. The highest value of *TreeSim* among *T1=T2*, *T1*$\sqsubseteq$ *T2*, and *T1*$\sqsupseteq$ *T2* is returned as the final similarity score.

For the example of Figure 5, 11 node-to-node correspondences, namely 7 equivalence and 4 abstraction/refinement relations, were identified by the matching algorithm. The biggest tree is *T2* with 13 nodes. Then, these are used to compute *TreeSim* between *T1* and *T2* by exploiting the above mentioned formula. In our example *TreeSim* is 0.58 for both *T1=T2* and *T1*$\sqsubseteq$ *T2*, which in turn is higher than the cut-off threshold of 0.5, and, therefore, the two trees globally match as expected and the correspondences connecting the nodes of the term trees are further used for data translation purposes.

PRELIMINARY EVALUATION

The approximate structure preserving matching algorithm has been implemented in Java (Giunchiglia *et al.*, 2007b). As the work in (Avesani *et al.*, 2005, Giunchiglia *et al.*, 2007c) indicates, it takes around one year to build a large scale evaluation dataset. We have already started the process of building such a dataset for the eResponse domain and extensive evaluation constitutes one of the key directions of our future work. Here, we only report our preliminary evaluation results based on the motivating example of Figure 5 and several dozens of similar test cases we have acquired so far. The reference results for these problems were established manually. Then, the results computed by our solution have been compared with the reference results.

Abstractions	Operation	Preconditions	$Cost_{T1=T2}$	$Cost_{T1\sqsubseteq T2}$	$Cost_{T1\sqsupseteq T2}$
$t_1 \sqsupseteq_{Pd} t_2$	***replace(a, b)***	$a \sqsupseteq b$; ***a*** and ***b*** correspond to predicates	1	∞	1
$t_1 \sqsupseteq_{D} t_2$	***replace(a, b)***	$a \sqsupseteq b$; ***a*** and ***b*** correspond to functions	1	∞	1
$t_1 \sqsupseteq_{P} t_2$	***insert(a)***	***a*** corresponds to predicate, function	1	∞	1
$t_1 \sqsubseteq_{Pd} t_2$	***replace(a, b)***	$a \sqsubseteq b$; ***a*** and ***b*** correspond to predicates	1	1	∞
$t_1 \sqsubseteq_{D} t_2$	***replace(a, b)***	$a \sqsubseteq b$; ***a*** and ***b*** correspond to functions	1	1	∞
$t_1 \sqsubseteq_{P} t_2$	***delete(a)***	***a*** corresponds to predicate, function	1	1	∞

Table 1. Abstractions, tree edit distance operations and their costs.

As match quality measures we have used *F-measure*. It varies in the [0-1] range. The version computed here is the harmonic mean of *precision* (the measure of correctness) and *recall* (the measure of completeness), namely that each of these was given equal importance (Euzenat and Shvaiko, 2007). While computing F-measure we considered only if the trees match globally. Based on our previous experience in (Giunchiglia *et al.*, 2007a) we used here a cut-off threshold of 0.5. As a performance measure we have used *time*. It estimates how fast our solution is when matching trees fully automatically. All these tests have been performed on a standard laptop: Core Duo CPU - 2Hz, with 2 GB of RAM, with the Windows Vista operating system, and with no applications running but a single matching system.

The average F-measure of the matching algorithm on the test cases we ran was 0.7, while the average execution time was 46ms. These results look encouraging, especially the execution time, though further extensive large scale evaluation is needed.

RELATED WORK

We discuss the related work along two dimensions: *(i)* ontology matching and *(ii)* ontologies in crisis management. Related to *(i)*, many different matching solutions have been proposed so far: see (Noy, 2004, Shvaiko and Euzenat, 2005) for recent surveys, while examples of individual approaches addressing the matching problem can be found on www.OntologyMatching.org. These solutions take advantage of the various properties of ontologies (e.g., labels, structures) and use techniques from different fields (e.g., statistics and data analysis, machine learning, linguistics). These solutions share some techniques and attack similar problems, but differ in the way they combine and exploit their results. A detailed analysis of the different techniques in ontology matching has been given in (Euzenat and Shvaiko, 2007).

The problem of discovery of web services on the basis of their capabilities has recently received a considerable attention. Most of the approaches to the problem of web service matching employ a single ontology approach, i.e., the web services are assumed to be described by the concepts taken from a shared ontology, see, e.g., (Paolucci *et al.*, 2002). The most similar to the solution that we used in our eResponse scenario, i.e. the one in (Giunchiglia *et al.*, 2007b), is the approach taken in (Aggarwal *et al.*, 2004) and in (Oundhakar *et al.*, 2005), where the services are assumed to be annotated with the concepts taken from various ontologies. The matching algorithm combines the results of atomic matchers that roughly correspond to the element level matchers exploited as part of the S-Match algorithm (Giunchiglia *et al.*, 2007a).

Related to *(ii)*, relevant works can be found within the DIP[3], ORCHESTRA[4], WORKPAD[5], SAFE[6], and CASCOM[7] projects. For example, in the DIP project, a GIS based on the web services modelling ontology has been developed (Tanasescu *et al.*, 2006). In WORKPAD, the key idea is that P2P information integration systems do not rely on a single ontology, but use mappings, dynamically established between peers, to collect and merge data from various sources when answering user queries in the crisis scenarios (Mecella *et al.*, 2006). The SAFE project aims at assisting in the management of the response to natural disasters. Specifically, the work in (Iannella *et al.*, 2007) reviews some of the emerging requirements for crisis information management systems and provides an outlook of the current and future technologies that are needed to address these requirements.

All these projects develop at design time domain ontologies for the crisis management. In turn, the matching solution we used can support coordination of web services at run time using both predefined ontologies and the "emerging" ontologies (being the LCC constraints) in order to search and chain web services at run time, which is crucial for many emergency situations.

CONCLUSIONS

We have presented a typical emergency response scenario, which includes geo-data sharing through geo-services provided by the GIS agencies and its formalization using LCC. The scenario is based on the current organizational model as used in Trentino region, Italy. We then discussed an automatic approximate structure preserving matching algorithm used as a solution to the semantic heterogeneity problem between different implementations of the required geo-services. We applied the matching algorithm to the eResponse web service matching scenario. The evaluation results, though preliminary, look encouraging.

Currently, the matching solution is elementary in the sense that it provides means to match web services available in the LCC interaction models. However, to run an interaction model, a peer should know which interaction model it wants to execute and with which peers it will be interacting. The ultimate goal is to provide a unifying framework based on interaction models that are mobile among peers, being a mechanism for web service composition and enabling ad hoc peer coalition formation, as required by hastily formed networks (Denning, 2006). In turn, future work on the approximate structure preserving semantic matching proceeds at least in the following directions: (*i*) conducting an extensive evaluation, and (*ii*) extending the matching approach for dealing with fully-fledged GIS ontologies.

[3] http://dip.semanticweb.org/
[4] http://www.eu-orchestra.org/
[5] http://www.workpad-project.eu
[6] http://nicta.com.au/research/projects/smart_applications_for_emergencies
[7] http://www.ist-cascom.org/

ACKNOWLEDGMENTS

This work has been supported by the OpenKnowledge[8] project (FP6-027253). We are grateful to Fausto Giunchiglia, Mikalai Yatskevich, Fiona McNeill and Paolo Besana for many fruitful discussions on the structure preserving semantic matching.

REFERENCES

Aggarwal R., Verma, K., Miller J.A., and Milnor W. Constraint driven web service composition in METEOR-S. In *Proceedings of IEEE SCC*, pages 23-30, 2004.

Avesani, P., Giunchiglia F., and Yatskevich, M. A large scale taxonomy mapping evaluation. In *Proceedings of ISWC*, pages 67-81, 2005.

Bernard, L., Craglia, M., Gould, M., and Kuhn W. Towards an SDI research agenda. In Proceedings of the *EC-GI & GIS Workshop,* 2005.

Denning, P. Hastily Formed Networks. *Communications of the ACM*, 49(4):15-20, 2006.

Euzenat J. and Shvaiko P. *Ontology matching*. Springer, 2007.

Giunchiglia F. and Shvaiko P. Semantic matching. *The Knowledge Engineering Review*, 18(3):265–280, 2003.

Giunchiglia F. and Walsh T. A theory of abstraction. Artificial Intelligence, 57(2-3):323-389, 1992.

Giunchiglia, F., Yatskevich, M., and Giunchiglia, E. Efficient semantic matching. In *Proceedings of ESWC*, pages 272–289, 2005.

Giunchiglia, F., M. Yatskevich, and P. Shvaiko. Semantic matching: Algorithms and implementation. *Journal on Data Semantics*, IX:1-38, 2007.

Giunchiglia, F., Yatskevich, M. and McNeill, F. Structure preserving semantic matching. In *Proceedings of the ISWC+ASWC Workshop on Ontology Matching*, pages 13-24, 2007.

Giunchiglia, F., Yatskevich, M., and Avesani, P. A large scale dataset for the evaluation of matching systems. In *Posters of ESWC*, 2007.

Groot R. and J. McLaughlin J. Geospatial Data Infrastructure: Concepts, Cases and Good Practice. *Oxford University Press*, 2000.

Iannella, R., Henricksen, K., and Rinta-Koski, O. Towards a Framework for Crisis Information Management Systems. In *Proceedings of TIEMS*, 2007.

Mecella, M., Angelaccio, M., Krek, A., Catarci, T., Buttarazzi, B., and Dustdar, S. WORKPAD: an Adaptive Peer-to-Peer Software Infrastructure for Supporting Collaborative Work of Human Operators in Emergency/Disaster Scenarios. In *Proceedings of CTS,* pages 173-180, 2006.

Miller A. G. WordNet: A lexical database for English. *Communications of the ACM*, 38(11), pages 39–41, 1995.

Nebert D.. Developing Spatial Data Infrastructures. The SDI CookBook, version 2.0. *GSDI-Global Spatial Data Infrastructure*, 2004.

Noy N.. Semantic integration: A survey of ontology-based approaches. *SIGMOD Record*, 33(4):65–70, 2004.

Oundhakar, S., Verma, K., Sivashanugam, K., Sheth, A., and Miller J.. Discovery of web services in a multi-ontology and federated registry environment. *International Journal of Web Services Research*, 2(3):1-32, 2005.

Paolucci, M. Kawamura, T., Payne, T., and Sycara K. Semantic matching of web services capabilities. In *Proceedings of ISWC*, pages 333-347, 2002.

Parent, C., Spaccapietra, S. and Zimanyi, E. *Conceptual modeling for traditional and spatio-temporal applications: MADS approach.* Springer, 2006.

Robertson D. A lightweight coordination calculus for agent systems. In *Declarative Agent Languages and Technologies,* pages 183-197, 2004.

Robertson D., Walton C., Barker A., Besana P., Chen-Burger Y., Hassan F., Lambert D., Li G., McGinnis J., Osman N., Bundy A., McNeil F. van Harmelen F., Sierra C., and Giunchiglia F.. "Models of interaction as a grounding for peer to peer knowledge sharing". *Advances in Web Semantics*, Vol. 1, 2007.

[8] www.openk.org/

Shvaiko P. and Euzenat J. A survey of schema-based matching approaches. *Journal on Data Semantics*, IV:146–171, 2005.

Tanasescu, V., Gugliotta, A., Domingue, J., Davies, R., Gutiérrez-Villarías, L., Rowlatt, M., Richardson, M., and Stinčić, S. A Semantic Web Services GIS based Emergency Management Application. In *Proceedings of ISWC*, pages 959-966, 2006.

Valiente G. *Algorithms on Trees and Graphs.* Springer, 2002.