Online Ontological Reasoning for Context-Aware Internet Services [1]

Alessandra Agostini and **Claudio Bettini** and **Daniele Riboni**[2]

The research group of the DaKWE laboratory at the University of Milan has been working for the last three years at the specification and implementation of a middleware – named *CARE*[3] – to support context-aware service adaptation for mobile users. *CARE* has three major goals: a) supporting the fusion and reconciliation of context data obtained from distributed sources, b) supporting context dynamics through an efficient form of reasoning, and c) capturing complex context data that go beyond simple attribute-value pairs.

While goal b) has been considered in other works [6, 11], it becomes more difficult to achieve when different sets of inference rules are provided by distributed sources. Even more difficult is to conciliate efficient reasoning with the expressiveness requirements imposed by the goal c).

The *CARE* middleware and its underlying technical solutions have been presented in [1, 3]. In our framework the contextual data, being by nature distributed, is managed by different entities (i.e., the user, the network operator, and the service provider). We call *profile* a subset of context data collected and managed by a certain entity. Each entity has a dedicated Profile Manager for handling its own context data. Profiles include both shallow context data and ontology-based context data which is expressed by means of references to ontological classes and relations. Both the user and the service provider can declare policies in the form of rules over profile data which guide the adaptation and final personalization of the service. A dedicated module is in charge of building the aggregated context data for the application logic. In particular, it evaluates adaptation policies and solves possible conflicts arising among context data and/or policies provided by different entities. The ad-hoc rule-based reasoner is particularly efficient if no ontological reasoning is performed, having linear complexity. Experimental results have shown that the evaluation of rules is executed in few milliseconds.

In our framework we need to model both simple context data such as device capabilities or current network bearer, and socio-cultural context data describing, for instance, the user current activity, the set of persons and objects a user can interact with, and the user interests. While the first category, that we call *shallow* context data, can be naturally modeled by means of attribute/value pairs, the second one calls for more sophisticated representation formalisms – such as ontologies – and we call it *ontology-based* context data. Similarly to other research works (e.g., [5] and [7]), we have adopted OWL [10] as the language for representing ontology-based context data. This choice

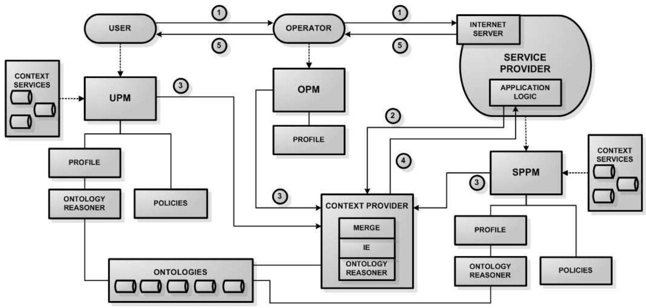

Figure 1. The *CARE* middleware architecture.

is motivated by the fact that the description logic languages underlying the *Lite* and *DL* sublanguages of OWL guarantee completeness and decidability, while promising high expressiveness. Moreover, a number of tools already exist for processing OWL ontologies and, being OWL a W3C Recommendation, the available utilities should further increase.

For a framework in which efficiency is a fundamental requirement, the introduction of ontological reasoning is particularly challenging. The hybrid approach implemented in *CARE* is based on a loose interaction between ontological and rule-based reasoning. While rule-based reasoning is performed at the time of the service request, ontological reasoning is mostly performed asynchronously by profile managers. However, in particular cases, ontological reasoning must be performed at the time of the user request, after having populated the ontology with instances collected from the distributed profile managers. In order to illustrate the hybrid mechanism, suppose that a user declared a policy rule asking to set her status to *busy* when involved in a business meeting:

$$\textbf{If } \textit{Activity} = \textit{`BusinessMeeting'} \textbf{ then } \textit{Status} = \textit{`Busy'} \quad (1)$$

Since the rule precondition predicate *Activity* is an ontology-based context parameter, its value must be inferred through ontological reasoning before evaluating the rule.

As an example, consider a possible definition of the *BusinessMeeting* activity:

$$\begin{aligned} \textit{BusinessMeeting} \equiv \textit{Activity} \sqcap \geq 2\,\textit{Actor} \sqcap \\ \forall \textit{Actor.Employee} \sqcap \exists \textit{Location.WorkLocation} \end{aligned}$$

---

[1] This work has been partially supported by Italian MIUR (FIRB ”Web-Minds” project N. RBNE01WEJT_005).

[2] DICo, University of Milan, via Comelico 39, I-20135 Milan, Italy, email: {agostini,bettini,riboni}@dico.unimi.it

[3] Context Aggregation and REasoning middleware.

Based on this definition, in order to check whether the user is involved in a business meeting it is necessary to have information about the people she is with (possibly derived by the user profile manager analyzing her agenda) and her current location (possibly provided by the network operator). This data is added to the assertional part of the ontology (i.e., the *ABox*).

Our initial experimental setup was based on the realization of the whole ABox upon receiving the context data from the profile managers. The current user activity was identified by performing nRQL queries to the well-known description logic reasoner Racer [8].

Even if OWL-DL guarantees completeness and decidability, performing online reasoning tasks with an OWL ontology could be computationally unfeasible, especially when providing an interactive service to a possibly huge number of users. Despite several assessments on the performance of reasoning with description logics are available, we performed some tests in order to verify the feasibility of executing ontological reasoning at the time of the service request with our specific OWL-DL ontologies. As expected, experimental results showed that query response times are strongly correlated to the number of instances of the examined ontology class as well as to the depth of the class within the ontology hierarchy. Our results confirmed that the execution of these ontological reasoning tasks at the time of the service request is unfeasible, even having a small ontology populated with few instances. In particular, query response times in our experiments are in the order of seconds.

We are investigating alternative approaches for overcoming the above mentioned computational issues. A possible solution consists in keeping the terminological part of the ontology (i.e., the *TBox*) static, in order to be able to perform the TBox classification [2] offline. In this way it is possible to save a good amount of computational time while serving user requests, since the ontology classification task is particularly expensive.

Furthermore, the assertional part of the ontology can be filled offline with those instances that are known *a priori*, i.e., before retrieving context data from the distributed profile managers. This data obviously depends on the particular domain addressed by the ontology. In the case addressed by our example, the ABox should be populated with a huge number of instances, including those that correspond to the employees of the user organization, and to particular locations (e.g., rooms belonging to the organization). After having populated the ontology with these instances, it is possible to perform the ABox realization [2] offline. Once again, ABox realization is an expensive reasoning task, which is unsuitable to perform online when the ontology contains a huge number of instances.

At the time of the user request, the ABox is filled with only those instances that are retrieved from the profile managers. Considering the ontology definition (1) of our example, the instances to be inserted into the ontology correspond to a new activity *currentActivity* – the one performed by the user – and to the relations that link that activity to its actors and location. Adopting this approach, the only reasoning task that must be performed online is the *instance checking* of the single *currentActivity* instance with respect to the *BusinessMeeting* concept.

As a preliminary step for assessing the feasibility of this approach, we are going to perform extensive experiments for estimating the execution times of this task in relation to various dimensions, including the TBox size, the number of instances that are known *a priori*, and the number of instances that are introduced into the ABox at the time of the user request.

Moreover, we are interested in testing some optimization techniques aimed at improving the efficiency of ABox reasoning. These optimizations are based on the use of relational database techniques. A well-known proposal in this sense is the InstanceStore system [9]. However, at the time of writing, InstanceStore has some limitations that are critical for our reasoning scenarios. Indeed, it does not allow the introduction of relations between individuals into the ABox. An alternative proposal for optimizing ABox reasoning by means of DBMS techniques can be found in [4]. Since in this case relations between individuals are supported, we are investigating the use of similar techniques in our framework.

REFERENCES

[1] A. Agostini, C. Bettini, N. Cesa-Bianchi, D. Maggiorini, D. Riboni, M. Ruberl, C. Sala, and D. Vitali, 'Towards Highly Adaptive Services for Mobile Computing', in *Proceedings of IFIP TC8 Working Conference on Mobile Information Systems (MOBIS)*, pp. 121–134. Springer, (2004).

[2] *The Description Logic Handbook: Theory, Implementation, and Applications*, eds., Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, Cambridge University Press, 2003.

[3] C. Bettini and D. Riboni, 'Profile Aggregation and Policy Evaluation for Adaptive Internet Services', in *Proceedings of The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous)*, pp. 290–298. IEEE Computer Society, (2004).

[4] Cui Ming Chen, 'Large Abox Store (LAS): Database Support for Abox Queries', Master Thesis, Concordia University, Montreal, Quebec, Canada, (September 2005).

[5] Harry Chen, Filip Perich, Timothy W. Finin, and Anupam Joshi, 'SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications.', in *Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous 2004), Networking and Services*, pp. 258–267. IEEE Computer Society, (2004).

[6] B. Grosof, 'Prioritized Conflict Handling for Logic Programs', in *Proceedings of the International Logic Programming Symposium (ILPS)*, pp. 197–211, (1997).

[7] T. Gu, X. H. Wang, H. K. Pung, and D. Q. Zhang, 'An Ontology-based Context Model in Intelligent Environments', in *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference*, (2004).

[8] Volker Haarslev and Ralf Möller, 'RACER System Description', in *Proceedings of Automated Reasoning, First International Joint Conference (IJCAR 2001)*, pp. 701–706. Springer, (2001).

[9] Ian Horrocks, Lei Li, Daniele Turi, and Sean Bechhofer, 'The Instance Store: DL Reasoning with Large Numbers of Individuals', in *Proceedings of the 2004 International Workshop on Description Logics (DL2004)*, volume 104 of *CEUR Workshop Proceedings*. CEUR-WS.org, (2004).

[10] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen, 'From SHIQ and RDF to OWL: The Making of a Web Ontology Language', *Journal of Web Semantics*, **1**(1), 7–26, (2003).

[11] R. Hull, B. Kumar, D. Lieuwen, P. Patel-Schneider, A. Sahuguet, S. Varadarajan, and A. Vyas, 'Enabling Context-Aware and privacy-Conscius User Data Sharing', in *Proceedings of the 2004 IEEE International Conference on Mobile Data Management*, pp. 187–198. IEEE Computer Society, (2004).