Modeling Utility Ontologies in Agentcities with a Collaborative Approach

Luigi Ceccaroni
Fujitsu Laboratories of America
595 Lawrence Expressway
Sunnyvale, CA 94085, USA
+1 408 530 4563
lc@fla.fujitsu.com

Myriam Ribiere
Motorola Laboratories
Espace technologique Saint Aubin
91193 Gif-sur-Yvette Cedex, France
+33 (0)1 69 35 48 39
myriam.ribiere@crm.mot.com

ABSTRACT
This paper presents experiences about the modeling and implementation of utility ontologies used within the Agentcities initiative. Utility ontologies include domain-independent concepts which most services developed within the project use. Ontology building was carried out collaboratively among very different partners from industry and academia. The application domain of the ontologies is an open, dynamic test-bed for agent deployment and they are explicitly designed to be shared by most services created within this environment. The ontologies are implemented in the DAML+OIL knowledge-representation language and a summary is given of the tools which currently let the user manage this language at a high level.

Keywords
Ontologies, Agentcities, Experimentation, DAML+OIL.

1. INTRODUCTION
Ontologies are being developed in AI to facilitate knowledge sharing and reuse. In general, ontologies can provide: (1) a shared and common understanding of a knowledge domain that can be communicated among agents and application systems; (2) an explicit conceptualization that describes the semantics of the data; (3) a basis for Web Services markup, facilitating their composition and mapping [3] [6]. Ontologies are considered to be a critical part of the work on the Semantic Web, which will allow software agents to communicate among themselves in meaningful ways [1], and attract attention not only from academic disciplines such as computer science, information science and artificial intelligence, but also from industries as diverse as the high-tech, financial, medical, educational and environmental sectors [4].

To obtain a shared and common understanding of a domain, a collaborative effort is necessary, involving ontology architects and domain experts; however, there are not many initiatives that have used and documented *collaboration* in *building ontologies*. Small-scale collaborations reflecting diverse viewpoints and backgrounds for the design of specific-domain ontologies exist (such as [5] and [2]), but participation in large ontology project is typically limited to academics coming from an AI background.

The European Commission funded Agentcities.RTD project is part of a worldwide initiative [8] designed to help and realize the commercial and research potential of agent based applications by constructing an open, distributed network of platforms hosting diverse agents and services. The ultimate aim of Agentcities is to enable the dynamic, intelligent and autonomous composition of services to achieve user and business goals. The Agentcities.RTD project includes 14 partners from academia and industry. Each partner deploys an agent platform, and agents and services based on that platform. The communication among these services has part of its semantic grounding in a series of *utility ontologies*, which model common, general concepts. Besides the utility ontologies, partners collaboratively designed several domain ontologies (which will be shared by and used within services) for the following domains: accommodation, geographic information, rating, restaurant, shows, transport and weather. A general service-interoperability ontology is also being modeled.

2. UTILITY ONTOLOGIES
In January 2002, a group of partners from the Agentcities.RTD project began modeling domain-independent concepts in the form of ontologies to be used by most services developed within the project. Identifying, descriptive and functional features of the four ontologies finally modeled (address, contact details, price, calendar) are presented in Table 1. During a meeting in February 2002, DAML+OIL[1] was chosen as the ontology modeling language, while FIPA-SL[2] was chosen as the content language. Although the DAML+OIL language is at the center of current research on the Semantic Web, there are drawbacks in using it: (1) the constant evolution of the language within the DAML project (the language is not yet stable); (2) available ontology editing tools (see section 2.2) are not satisfactory and do not handle all the features of the language, which makes them not apt to be used for the complete cycle of ontology design and implementation; (3) there is not much documentation on experience and good practices in using DAML+OIL to build usable and reusable ontologies.

2.1 Knowledge acquisition
International standards were taken into accounts when modeling the utility ontologies, though none of them was sufficiently concise to be fully adopted by the short-term EU Agentcities project. The ontology specifications developed within Agentcities

[1] See [http://www.daml.org/language/].

[2] See [http://www.fipa.org/specs/fipa00008/].

therefore differ from the ontologies implied by existing standards, but they are in no way intended to create separate definitions for concepts defined by standards bodies. We indeed are working towards a convergence of the *ContactDetails* ontology with the vCard standard[3] and of the *Calendar* ontology with the iCalendar standard[4].

2.2 Ontology editors

There are, at the moment, a number of more or less generic editors to create and manage ontologies, but just a few of them can manage the DAML+OIL language. To the best of our knowledge, there are only two ways to carry out this management process at a high level, neither of which is very practical or satisfactory:

1. OilEd and Protégé-2000

- *Creating*: any program that can save files as RDFS, for example (with some limitations) the OilEd[5] editor.
- *Editing*: Protégé-2000[6] with the Ontoviz graphical visualization plug-in (or other equivalent plug-ins).
- *Exporting*: OilEd, which (with some limitations) can import RDFS files that have been edited in Protégé-2000.

2. Ontolingua and Chimaera

- *Creating*: any program that can save files as DAML+OIL.
- *Editing*: Ontolingua environment. To import a DAML+OIL file into the KIF-based Ontolingua, it is necessary to use Chimaera[7].
- *Exporting*: Chimaera (with some limitations and a user unfriendly interface).

We did not extensively test yet any ontology consistency-checking and reasoning tools, available for these methodologies, such as JTP and FaCT.

In conclusion, we acknowledge that, if we had not required an XML-based language as the ontology language, an alternative, more practical solution to ontology management would have been to use only the Ontolingua environment and to work with KIF ontologies, thus avoiding a number of language translations.

3. COLLABORATIVE APPROACH

Researchers taking part in the Agentcities.RTD project come from very different areas of study and have different perspectives on ontology modeling, but, significantly, they pledged to adopt the same ontological commitment. That is, they agree to adopt common, predefined ontologies when communicating about a domain of interest or to express general categories, even if they do not completely agree on the modeling behind the ontological representations. Where ontological commitment is lacking, it is difficult to converse clearly about a domain and to benefit from knowledge representations developed by others. The ongoing development of the utility ontologies proceeds with an eye towards ensuring that their future users will find their characterizations to be sufficiently correct, clear and concise. Ontological commitment is thus an integral aspect of ontological engineering [5] in the Agentcities.RTD project.

Collaborative development of ontologies in Agentcities was carried out through both face-to-face meetings and remote communication (email and IRC sessions). No satisfactory on-line tool or environment exists that supports the DAML+OIL language and collaborative development.

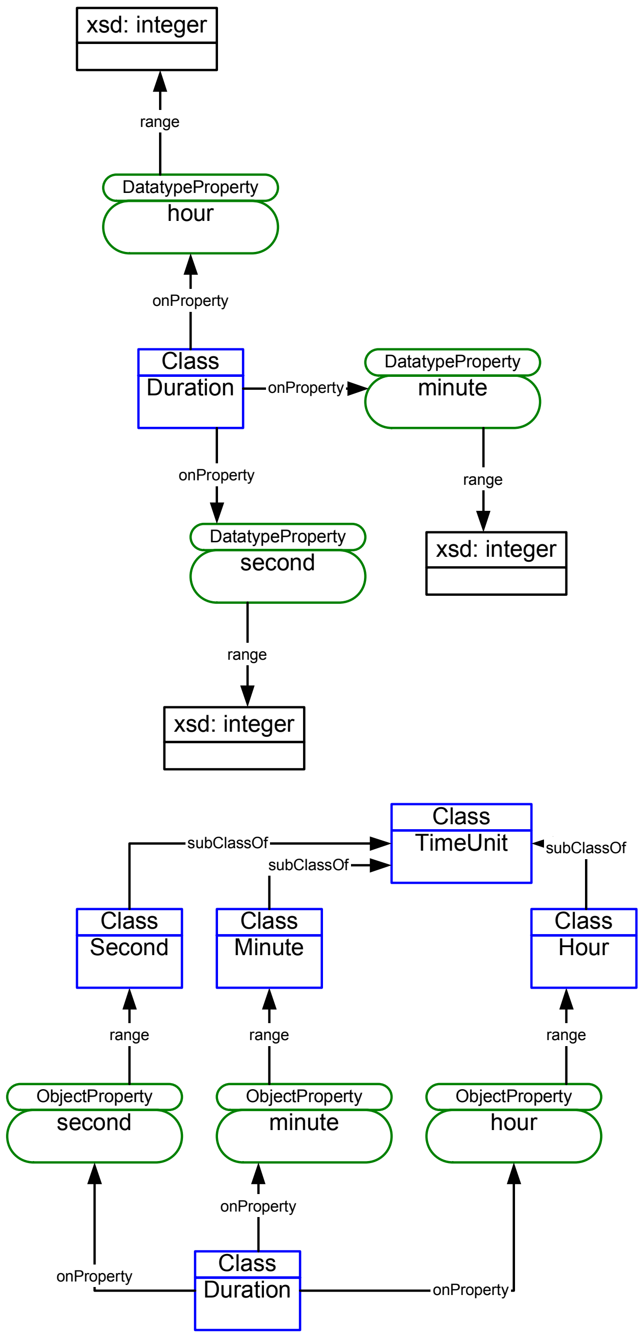

Figure 1. Methods of representing the range of properties.

[3] vCard 3 is defined by RFC 2426 [http://www.imc.org/pdi/].

[4] iCalendar is defined by RFC 2445 [http://www.imc.org/pdi/].

[5] See [http://oiled.man.ac.uk/index.shtml].

[6] See [http://protege.stanford.edu/].

[7] See [http://www.ksl.stanford.edu/software/chimaera/].

3.1 Methodology

The construction of ontologies is a time-consuming and complex task, in particular during the conceptualization phase, when developers define the set of concepts and their relations by an intermediate representation often based on tabular and graphical notations. A common graphical representation has to be agreed and a common media for the interchange of proposals and a decision system to overcome disagreements have to be chosen.

In Agentcities, during the conceptualization phase, the following issues had to be dealt with. We acknowledge that the very classification of these issues is subjective and that it is not the only possible one.

Data types* versus *classes. As shown in Figure 1, there are two ways of representing the range of properties: as a predefined data type (for example, integer; above in the figure) or as a class (for example, subclass of *TimeUnit*; below in the figure). Using classes is semantically richer, but more complex.

Individuals* versus *classes. There are two ways of representing the elements of a class: as individuals or as subclasses. Using classes is semantically richer and makes the extension of ontology easier. Even if more complex, in general the use of classes was preferred.

Properties of properties. As shown in Figure 2, there are 3 ways of representing properties of other properties. In the example, we want to represent the kind (e.g., personal or business) of properties of the *ContactDetails* class, such as phone number and pager[8]. One possible way to achieve this is to define a property for each, which has as the range a common concept called *ContactDetailType* (top part of the figure). In this option, as well as in the next one, we acknowledge the fact that the notions of personal/business and private/work are common to many concepts, and we exploit it to simplify the design. The *ContactDetailType* class has thus three individuals, *PersonalWork*, *PersonalPrivate* and *Business*, which are the possible values of the range of the *phoneNumberType* and *pagerType* properties (or, in other terms, the possible types of *phoneNumber* and *pager*). A second possibility, to avoid defining a property of a property (which some languages do not allow), is to introduce bridge classes as the range of *phoneNumber* and *pager* (central part of the figure). In our modeling, these first 2 approaches are semantically equivalent and interchangeable. A third possibility is to have specific subclasses, representing the different type for each property of *ContactDetails* (bottom part of the figure). For example, for the *PhoneNumber* class, we define explicitly all the different subclasses: *PhoneNumberBusiness*, *PhoneNumberPersonalWork*, and *PhoneNumberPersonalPrivate*. In general, we think that the creation of additional classes is preferable only in the case in which the resultant representation is semantically richer.

Cultural differences. Even though the concepts included in the utility ontologies are very general, the differences in the cultural background of each partner caused some discrepancies in the design of the ontologies, in particular, in the case of the address ontology. Apart from the most general level, different countries use different conventions to express an address and thus generalization is not easy.

[8] Other (not shown) properties of *ContactDetails* which behave in the same way are: *mobile phone number*, *web page*, *fax number*, *email*, and *other*. Two other properties of *ContactDetails* which have a different behavior are: *name* and *address*.

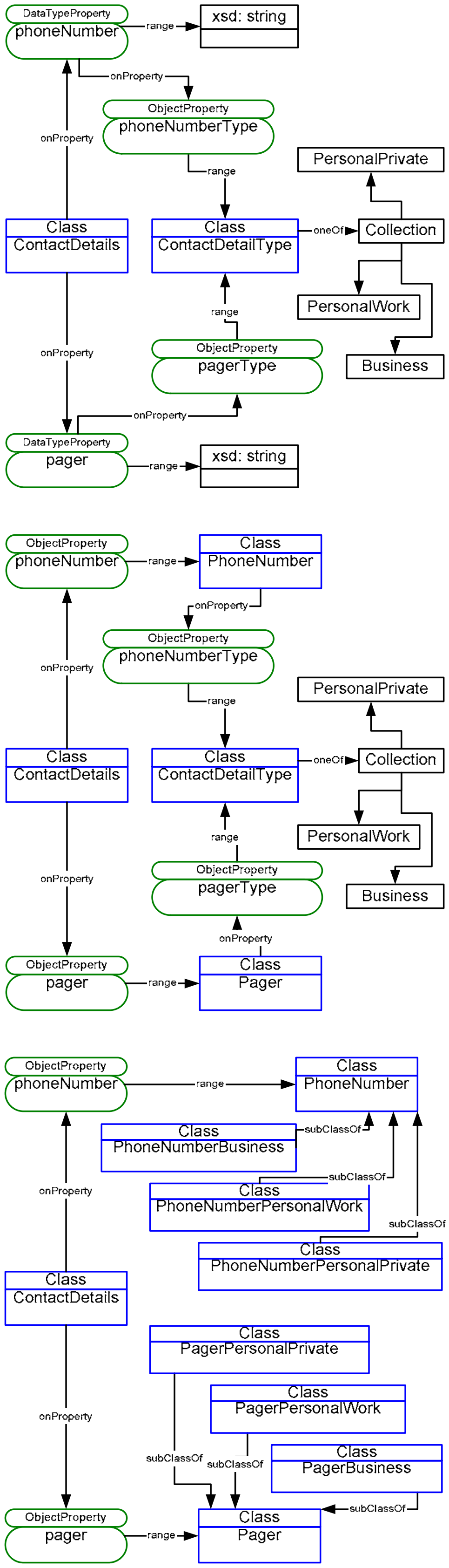

Figure 2. Methods of representing properties of properties.

Table 1. Features of the four utility ontologies.

	Address	Contact details	Price	Calendar
Name	*Address.daml*	*ContactDetails.daml*	*Price.daml*	*Calendar.daml*
Subject	Management of most types of addresses of common use.	Management of contact details for a person or for a business.	Management of prices.	Management of events in time.
List of higher-level concepts	Address, BuildingSubDivisionType, PublicPlace	ContactDetails, ContactDetailType, Name	Price, PriceRange	Calendar, Date, DayOfWeek, Duration, Time, TimeFormat
Integrated ontologies	none	Address ontology	none	none
Number of classes	13	5	6	6
Number of instances	0	3	0	9
Number of properties	18	27	4	15
Number of class at 1st, 2nd and 3rd level	3, 10, 0	3, 2, 0	2, 4, 0	6, 0, 0
Number of class leaves	10	4	5	6
Average branching factor	3	1	2	0
Average depth	2	1	2	1
Highest depth level	2	2	2	1

4. CONCLUSIONS

Four *utility ontologies* for the common, general concepts of Address, Contact Details, Price and Calendar have been created. These ontologies have been modeled through a collaborative effort among several partners of the EU Agentcities.RTD project. The modeling process took into account all the available, compatible indications on methodology coming from the *ontology community* and this paper enriches those indications through extensive practical experience. The utility ontologies described here are the manifestation of a shared understanding and will be used, within the Agentcities network, as part of the semantic grounding for the communication among Web Services. The implementation language of the ontologies is DAML+OIL

5. ACKNOWLEDGMENTS

The authors wish to extend their thanks to the other partners in the EU Agentcities.RTD project involved in the modeling phase, and to Jonathan Dale for the feedback after having read a preliminary version of this paper. The research described in this paper is partly supported by the EC project Agentcities.RTD (IST-2000-28385). The opinions expressed in this paper are those of the authors and are not necessarily those of the EU Agentcities.RTD partners.

6. REFERENCES

[1] Berners-Lee, T., Hendler, J., and Lassila, O. *The Semantic Web*. In Scientific American, 284, 5, pp. 34-43, 2001.

[2] Ceccaroni, L. *OntoWEDSS - An Ontology-based Environmental Decision-Support System for the management of Wastewater treatment plants*. Ph.D. thesis, Universitat

[3] Fensel, D., Horrocks, I., Van Harmelen, F., Decker, S., Erdmann, M., and Klein, M. *OIL in a nutshell*. In R. Dieng et al. (editors) *Knowledge Acquisition, Modeling, and Management*, Proceedings of the European Knowledge Acquisition Conference (EKAW2000), Lecture Notes in Artificial Intelligence (LNAI) , Springer-Verlag, pp. 1-16, 2000.

[4] Gruninger, M., and Lee, J. *Ontology applications and design*. In Communications of the ACM, 45, 2, pp. 39-41, 2002.

[5] Holsapple, C.W., and Joshi, K.D. *A collaborative approach to ontology design*. In Communications of the ACM, 45, 2, pp. 42-47, 2002.

[6] McIlraith, S.A., Son, T.C., and Zeng, H. *Mobilizing the Semantic Web with DAML-Enabled Web Services*. In Proceedings of Autonomous Agents 2001 – Ontologies in Agent Systems (OAS 2001) workshop (Montreal, Canada), pp. 1-11, 2001.

[7] Uschold, M., and Gruninger, M. *Ontologies: principles, methods and applications*. In The Knowledge Engineering Review, 11, 2, pp. 93-136, 1996.

[8] Willmott, S., Dale, J., Burg, B., Charlton, P., and O'Brien, P. *Agentcities: A Worldwide Open Agent Network*. In The Agentlink Newsletter, 8, pp. 13-15, 2001.