LEARNING SERVICES BASED ON FORMAL CONCEPT REASONING

Gonzalo A. Aranda-Corral
Dpto. de Ingeniería Electrónica, Sistemas Informáticos y Automática-Universidad de Huelva
Ctra. Huelva-Palos, s/n. 21819 Palos de la Frontera. Huelva. España

Joaquín Borrego-Díaz, Antonia M. Chávez-González
Dpto. de Ciencias de la Computación e Inteligencia Artificial- Universidad de Sevilla
E.T.S. Ingeniería Informática, Avda. Reina Mercedes s/n. 41012-Sevilla España

ABSTRACT

A formal foundation of automated service discovering for Semantic Web is proposed. The approach is based on the formalization of the problem using an agent oriented programming language (ConGolog), as well as on the use of the Formal Concept Analysis as a tool for knowledge extraction.

KEYWORDS

Ontological Engineering, Semantic web services, Formal Concept analysis, Situation Calculus, ConGolog.

1. INTRODUCTION

A need for agent interoperability -in a MultiAgent System (MAS)- is a common domain's conceptualization and a shared vocabulary for exchanging information, that is ontologies (Sycara and Paolucci 2004). The aim of the envisioned Semantic Web (SW) (Berners-Lee and Lassila 2001) is to satisfy these needs, including the common understanding of Semantic Web Services (SWS) in order to allow their discovery, composition and invocation (Lara et al. 2003).

Several ontologies to specify SWS exist, as OWL-S (http://www.w3.org/Submission/OWL-S) or WSMO (http://www.wsmo.org). The idea is that any feature of the SWS must be logically specified, mainly properties, capabilities, interfaces and effects. Moreover, it is also necessary to provide automated reasoning technologies which are fully justified on logical soundness (Benatallah et al. 2005).

1.1 An Approach to Ontology Evolution Induced by MAS

Recently, several new multiagent toolkits are oriented to strongly knowledge-based agents (see e.g. NUIN, Dickinson and Wooldridge 2005). The successful use of this kind of systems depends on the availability of robust and efficent reasoning with the logic on which they are based. Actually, the main challenge is to grant that any agent's activity has logical legitimacy. Only in this way the systems can help users (explaining why an application behaved in an unexpected manner, Horrocks 2005) and designers (finding logical errors in the system).

Another important challenge that MAS could face, in the future, will be to support ontological transformations, induced by the execution of MAS itself. This is a critical issue, because ontology revision is expensive, it may even be dangerous for companies. Therefore, the provision of semi-automated ontology evolution mechanisms may be a considerable point of the field of knowledge-based MAS. Note that such evolution will depend on data stream and cognitive decisions made by engineers, among other reasons. On

the one hand the MAS running can suggest, the transformation of capabilities of services, although on the other hand, the logical complexity of this revision may be very high.

The aim of this paper is to propose an approach for ontology revision based on the reasoning with ontological models associated to MAS services (which is feasible when the services structure is not too complex), and under a *referee* supervision only to select interesting new services. The structure of the paper is as follows. Once discussed the impact of service learning in ontologies involved in MAS (subsection 1.2), the subsection 1.3 is devoted to commenting relevant issues of Web service formalization, from the point of view of automated reasoning. In section 2 a running example is presented. The method for ontology discovering is presented in section 3. The paper finishes with some comments on related work and future issues.

1.2 Structure of Potentially New Services

In this paper we propose to describe the formal foundations for a method which extracts a simple structure, composed of potentially useful new services, as well as how to create new services from this structure. This creation can be considered as an ontology revision process and preserves some compatibility with the original ontology and certain fundamental properties.

Despite such evolution may be critical (for example, when it affects to goal descriptions), current MAS do not provide tools to manage them. In fact, it could be interesting to have an agent able to offer the learned evolution.

The logical specification of a method for ontology revisions might avoid some undesirable effects which ontology evolution can produce, from the point of view of automated reasoning (Alonso et al. 2006). Roughly speaking, the studied option in this paper is that the ontological engineer, assisted by a theorem prover, proposes a revision which must be accepted by the user. Generally, the changes will be based on the analysis of a given *argument* by theorem provers (in our case, arguments and executions are equivalent because ConGolog is interpreted by PROLOG). Finally, in order to certify logical soundness, the revision is based on a formal framework where the revision can be certified by means of a model finder and a theorem prover (Borrego-Díaz and Chávez-González 2005, Horrocks 2005).

1.3 Automated Reasoning on Services

Following the approach of (Keller et al. 2004), a goal is a pair of formulae, $G = (\phi^{post}, \phi^{ef})$ describing the post-condition (information state) and the effect (on the world) of them. The capability is described by a 4-uple $C = (\psi^{pre}, \psi^{as}, \psi^{post}, \psi^{ef})$ which specifies the preconditions on input, the information about the state of the world, the output properties and the effect on the world. Finally, the dynamic nature of services makes the formal representation of state change advisable. J. McCarthy's Situation Calculus (SC) is a formalization of state change, useful to reason on rational agents (Reiter 2001). Agent oriented language ConGolog (de Giacomo et al 2000), based on SC, is useful to specify properties of SWS (McIlraith and Cao 2002, Martínez and Lespérance 2004). Examples of SC specifications are shown in figure 1.

The service σ has the capability C :

$$\Omega \models \forall s[Holds(\psi^{pre} \wedge \psi^{as}, s) \rightarrow Holds(\psi^{post} \wedge \psi^{ef}, Do(\sigma, s))]$$

The service σ provides the goal G :

$$\Omega \models \forall s Holds(\psi_{\sigma}^{pre} \wedge \psi_{\sigma}^{as}, s) \rightarrow Holds(\psi_{\sigma}^{post} \wedge \psi_{\sigma}^{ef}, Do(\sigma, s)) \wedge Holds(\phi_{G}^{post} \wedge \phi_{G}^{ef}, Do(\sigma, s))$$

Existence of a service that provides the goal G :

$$\Omega \models \exists \sigma \forall s[Holds(\psi_{C}^{pre} \wedge \psi_{C}^{as}, s) \rightarrow Holds(\psi^{post} \wedge \psi^{ef}, Do(\sigma, s))]$$

Figure 1. Partial specification of provision problems in the Situation Calculus

If Ω denotes the system specifications, where the ontologies would be included, the fact that σ has capability C ($\sigma \models_{\Omega} C$) is formalized in the first condition of figure 1. That is, when the action term is

executed, the post-condition is satisfied (in the final state), as well as the corresponding effects. The second example is similar. Finally, the existence of services is actually a planning problem in SC.

SWS are closely related to our ability to operate with robust reasoners on the SW, as well as with Knowledge Representation formalisms. By representing the SWS in SC, several tasks of SWS engineering are generalized in a direct way. For example, goal provision checking usually based on matching (between effects/postconditions of the service/goal specification), can be generalized using logical entailment.

Analogously, service discovering can be generalized to planning. Nevertheless, this generalization is based on backward reasoning: it starts with goal specification. A more ambitious generalization method is based on the analysis of MAS executions, in order to learn new services. Learned services have to be specified by means of high order planification methods, as in ConGolog, which is useful to learn SWS (McIlraith et al. 2001, McIlraith and Cao 2002). Since such language is based on a simple solution to the frame problem (Reiter 2001), which is interpreted by PROLOG, it is possible to identify executions with provability. The main difficulty of this approach is its logical complexity. Our approach is based on Formal Concept Analysis (FCA) (Ganter and Wille 1999), easy to use, although it will not provide ConGolog specification of the new service.

2. AN EXAMPLE

We describe an example where we will carry out the basic steps to create a new service and to revise the ontologies associated, adapted from an example from (de Giacomo et al. 2000). Let it be considered a market with computers and accessories. Under the point of view of the ConGolog seller agent, there are several exogenous actions, but to reduce in complexity, we suppose that a single exogenous action exists, *newClient(cid,x)*, which means that the client identified by *cid* asks for the article x. Moreover, the seller agent has the action *serve(cid,x)* (to supply the article x to client *cid*). When *newClient(cid,x)* is executed, the fluent *ClientWaiting(cid,x)* becomes true and, in this case, the system has to accept the client, *acquire(cid)*, and to serve him the article x. Part of the seller agent code is shown in figure 2.

Precondition of *acquire(cid)*: $Poss(acquire(cid), s) \equiv Holds(ClientWaiting(cid, x), s).$

Effect of action *newClient(cid,x) on ClientWaiting(cid,x):* $Holds(ClientWaiting(cid,x),do(a,s)) \equiv$
$\equiv a=newClient(cid) \vee Holds(ClientWaiting(cid,x),s) \wedge a \neq acquire(cid).$

Seller agent: $[\pi\, cid, x.acquire(cid, x); serve(cid, x)]^{\|}; \neg\exists cid.ClientWaiting(cid, x)?$

Figure 2. Part of ConGolog seller agent

According to the seller agent, a non-deterministic number of processes are serving products to clients. A final state is reached when there are no more clients waiting. Some executions are depicted in figure 3 as formal context (considering executed actions as attributes). Let the articles supplied by the seller be:

PC (computer), WXP (operating system), PSB (USB memory) and M (monitor).

	Client	*newClient*	*acquire*	*servePC*	*serveWXP*	*servePSB*	*serveM*
*Exec.*1	*c*1	X	X	X	X	X	X
	*c*2	X	X				X
	*c*3	X	X	X	X		
*Exec.*2	*c*4	X	X	X	X	X	X
	*c*5	X	X		X	X	X
*Exec.*3	*c*6	X	X	X	X		
	*c*7	X	X				X
	*c*8	X	X	X	X		

Figure 3. Action traces of some executions of the MAS as a formal context

3. DISCOVERING NEW SERVICES: CONCEPT SERVICES

In order to discover new services from data, we assume the cognitive principle that an implicit conceptual structure exists on data and services. Formal Concept Analysis (FCA) is a mathematical theory that allows to extract such a structure. FCA is a way to mathematize *concepts* and *conceptual hierarchies* (Ganter and Wille 1999). In Artificial Intelligence, FCA is a knowledge representation tool, useful for Intelligent Data Analysis and Knowledge Acquisition (KA). It also represents a formal framework for implication and association rules. The procedures involved in FCA have nice computer implementations as the ConExp system (http://sourceforgenet/projects/conexp) for the basics, and other advanced and specific tools exist.

An interesting application of the discovering process is that an ontology can be induced from the services. The ontology induced from the formal context is given in figure 4(left).

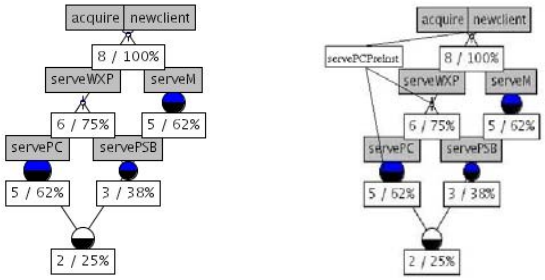

Figure 4. Lattice structure of concept services induced by executions of figure 3 (left) and the selected extension (right).

Each lattice node is called a *concept service*. A concept service is a concept which is extracted from the formal context in FCA sense. That is, a pair *(Act,Obj)* where *Act* is a set of actions (intension of concept service) such that *Obj* (extent) is the set of executions in which every action of *Act* is executed, and the set *Act* is the set of common actions of every execution in *Obj*. Thus, concept services are sets of services, that formal context of executions suggest to integrate. The subset relation is understood as "subtask of", that is, $\alpha \xrightarrow{super} \beta$ means

$$\forall s_1, s_2 (s_2 = Do(\alpha, s_1) \rightarrow \exists s_3 \leq s_4 \leq s_2 (s_1 \leq s_3 \wedge s_4 = Do(\beta, s_3))$$

That is, the execution of the first action term implies the execution of the second one. Initially, a concept service is not a new service. It is possible that the concept intension can be directly composed, or it can occur that it is unfeasible (or too expensive). Finally, the user is the one who have to decide which concept services must be considered, according to cost criterions, for example, a deep analysis of the conceptual structure can provide the learning of new services. Following this proposal, consider the formal context of figure 3 . It is possible to compute a *stem basis* associated to the formal context (Ganter and Wille 1999). A stem basis is, in propositional Horn logic terms, a non-redundant complete implicational set. Moreover, it is valid in every execution described in the context. From the context of figure 3, the following stem basis is obtained:

$$\begin{cases} newclient \ . \\ acquire \ . \\ newclient \ \wedge acquire \ \wedge servePSB \ \rightarrow serveWXP \\ newclient \ \wedge acquire \ \wedge servePSB \ \rightarrow serveM \\ newclient \ \wedge acquire \ \wedge servePC \ \rightarrow serveWXP \end{cases}$$

From this stem basis, every valid implicational knowledge in the formal context can be deduced. The fifth one can be purged, based on redundancy analysis, to yielding $servePC \rightarrow serveWXP$. Suppose that orders for *PC's* are made before that orders for *WXP* (when both articles are ordered),

$$\Omega \ \models Holds(serve(u, PC), s) \rightarrow \exists \alpha Holds(serve(u, WXP), Do(\alpha, s)) \quad (\dagger)$$

This fact suggests to offer a new service (*to serve a* PC *with* WXP *preinstalled*) that must contain

$$[serve(u, PC); serve(u, WXP)] \qquad (*)$$

It will be denoted by $servePCPreins(cid)$. The precondition for the new service can be conjectured as:

$$Poss(servePCPreins(cid)) \equiv Holds(ClientWaiting(u, PC), s) \wedge$$
$$\wedge Holds(ClientWaiting(u, WXP), s)$$

Next, the seller agent should be informed of the new service, as well as its specification. For example, it is possible that the seller agent accepts the new service based on cost criterions. Let us suppose that it has a functional fluent $cost(\alpha, s)$ to estimate the cost of service α in the state s. If it is provable, from Ω, that the cost of the new service is lower, then the new service is accepted. However, before considering the above preconditions as definitive, it is necessary to consider another aspect: the specification of the new service should provide similar properties to other concept services. That is, it is necessary to achieve a sound ontological insertion of the new service in the concept services lattice. This is essentially a step of ontology evolution assisted by automated reasoning systems. It is necessary to use data to select the best extension by insertion of the structure, for example, evaluating the cognitive entropy of the possible extensions (Borrego-Díaz and Chávez-González 2005b). The method to make the selection is fully automatized, if ontology is *Lattice Categorical* (L.C.) (Borrego-Díaz and Chávez-González 2005). Roughly speaking, a L.C. theory proves the lattice structure endowed by its concepts.

For this running example, *servePCpreinst(.)* must be inserted in the concept services lattice. The insertion is obtained according to the method shown in (Borrego-Díaz and Chávez-González 2005). This is semiautomatic, with human intervention in the specification given in the last step only. We outline the main steps of the process. We suppose that Ω is lattice categorical with respect to the ontology of figure 4(left).

Step 1: ***Conjecture on the new ontology*****:** $\Omega + \{ \; servePCPreinst(cid) \rightarrow serve(cid, PC) \; \}$

Step 2: ***Find extensions of the lattice*****.** The theory above is not L.C. The model finder MACE4 (http://www-unix-mcs.anl.gov) lists three L.C. ontological extensions by insertion of the new element.

Step 3: ***Select the extension to specify the service*****.** Afterwards, the entropy variation between the original ontology and the new one (with respect to data of figure 3) is computed. The extension which induces lower entropy variation (estimated by Shannon's index, Borrego-Díaz and Chávez-González 2005b) is selected. In our example, the selected extension is the third one. In terms of SC:

Theorem: *If Ω is lattice categorical with respect to the ontology of MAS concept services, then*

$$\Omega + \{(servePCpreinst(cid) \xrightarrow{\text{super}} serve(cid, x)) \leftrightarrow (x = PC \vee x = WXP)\}$$

is a lattice categorical extension of Ω with respect to the third ontological extension.

Step 4: ***High order planning of the new service*****.** The conjecture (*) as the new service is valid if (†)is entailed by Ω. This strongly depends on Ω. For example, if the sentence (†) is not provable from Ω, the concept service *servePCPreinst(cid)* should be programmed in ConGolog as

$$[serve(u, PC); serve(u, WXP)] \,||\, [serve(u, WXP); serve(u, PC)]$$

4. CONCLUSION

This paper is a formal approach to learning services based on FCA. Future works will be focused on the implementation of the method on a MAS platform as JADE (with limited ontological expressiveness).

An advantage of ConGolog specifications of SWS is that one can use the PROLOG interpreter to obtain big sets of simulations. Since the information obtained from simulation is potentially infinite, the creation of new services from the formal context associated to big sets of executions is accurate. Finally, ontology revision can produce discrepancies if agents are not aware of changes.

An important feature that has not been considered throughout this paper is the role of performatives in MAS communication. This dimension is necessary in ontologies on services oriented to SW (Gibbins et al. 2004). It is feasible to insert these features in ConGolog, although they make it more difficult to compute new *commonsense knowledge* generated by the MAS, that is, the revision of ontologies.

ACKNOWLEDGEMENTS

This work is partially supported by the project *Verified Systems for Semantic Web Reasoning* (TIN 2004-03884), Spanish Ministry of Education and Science, cofinanced by FEDER founds.

REFERENCES

Alonso-Jiménez J.A., Borrego-Díaz J., Chávez-González A.M. and Martín-Mateos F.J., Foundational Challenges in Automated Data and Ontology Cleaning in the Semantic Web, *IEEE Intelligent Systems*, 21(1):42-52 (2006).

Benatallah B., Hacid B.S., Leger A., Rey C., Toumani F., On automating Web services discovery, *Very Large Database Journal* 14:84-96 (2005).

Berns-Lee T., Hendler J. and Lassila O., The Semantic Web, *Scientific American* 284(5):34-43 (2001).

Borrego-Díaz J. and Chávez-González A.M., Extension of Ontologies Assisted by Automated Reasoning Systems, *Proc. 10th Int. Conf. on Computer Aided Systems Theory, EUROCAST'05*, LNCS 3643, Springer, 2005.

Borrego-Díaz J. and Chávez-González A.M., Controlling Ontology Evolution Through Cognitive Entropy, Proc. of ISWC-2005 Workshop on Uncertainty Reasoning for the Semantic Web (2005).

Dickinson I. and Wooldridge M., Some Experiences with the Use of Ontologies in Deliberative Agents in *Ontologies for agents: Theory and Experiences*, Birkhäuser, pp.277-298 (2005).

Feier C. and Domingue J., *WSMO Primer*, Draft, DERI Institute, http://www.wsmo.org/TR/d3/d3.1/v0.1/

Horrocks I., Applications of Description Logics: State of the Art and Research Challenges, *Proc. of Int. Cong. on Conceptual Structures ICCS'05*, LNAI 3596, Springer, pp. 78-90, 2005.

Keller U, Lara R., Polleres A., Lausen H., *Inferencing Support for Semantic Web Services: Proof Obligations*, WSML Deliverable D5.1 v0.1, http://www.wsmo.org/2004/d5/d5.1/v0.1/20041112/

Ganter B. and Wille R., *Formal Concepts Analysis. Mathematical Foundations* Springer, Berlin (1999).

de Giacomo G., Lespérance Y. and Levesque H.J., ConGolog, a concurrent programming language based on the situation calculus, *Artificial Intelligence* 121(1-2):109-169 (2000).

Gibbins N., Harris S. and Shadbolt N., Agent-based Semantic Web Services, *Journal of Web Semantics* 1(2):141-154 (2004).

Lara R., Lausen H., Arroyo S., de Bruijn J. and Fensel D., Semantic Web Services: Description Requirements and Current Technologies, *Int. Workshop on Electronic Commerce, Agents, and Semantic Web Services* (2003).

McIlraith S. A. and Cao Son T. and Zeng H., Semantic Web Services, *IEEE Intelligent Systems* 16(2): 46-53 (2001).

McIlraith S.A. and Cao Son T., Adapting Golog for Composition of Semantic Web Services, *Proc. 8th Int. Conf. Principles and Knowledge Representation and Reasoning KR 2002*, pp. 482-496, Morgan Kaufmann (2002).

Martinez E. and Lesperérance Y., IG-JADE-PKSlib: An Agent-Based Framework for Advanced Web Service Composition and Provisioning, *Proc. AAMAS'04 Workshop on Web-services and Agent-based Eng.*, pp. 2-10 (2004).

Reiter R., *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*, MIT Press, 2001.

Stumme G., TaouilR., Bastide Y., Pasquier N. and Lakhal L.. Computing Iceberg Concept Lattices with Titanic. *Data & Knowledge Eng.* **42**, 189--222 (2002).

Sycara K., Paolucci M., Ontologies in Agent Architectures, in *Handbook on Ontologies* (S. Staab and R. Studer eds.), Springer, pp. 343-364, 2004.

Bellifemine F.L., Caire G., Greenwood D.. *Developing Multi-Agent Systems with JADE*. Wiley Series in Agent Technology, 2007.