Agents and Web Services

Stefania Costantini[1]

Dip. di Informatica, Università di L'Aquila, Coppito 67100, L'Aquila, Italy
stefcost@di.univaq.it

Abstract. This is a position paper that comments about the role of agents and, in particular, of computational logic agents in the realm of Semantic Web, which is an important field of application of Artificial Intelligence results, techniques and technologies.

1 Introduction

The starting point for the considerations proposed below is a recent editorial by Terry Payne on IEEE Intelligent Systems [1] where a comparison among agents and web services is proposed. At first sight, web services can be seen as a very limited kind of agent, where some distinctive features of "intelligent" agents are not only not required, but even potentially harmful. As they are, however, web services show many limitations and actually they might be empowered, in the perspective of a Semantic Web and of Web 3.0 applications, by some more "agency". In the same issue, Xuan Shi [2] emphasizes that, at present, the discovery and use of web services is not a trivial task, as different interfaces may define the same service, and similar interfaces may define different services. According to him, this kind of ambiguities might be managed only by adding more "intelligence" and even some kind of "cognitive psychology" to cope with the "cognitive disorder" that even machines and systems may experiment as their "intelligence" grows. Shi also notices that the semantic of web services actually lies in the specification, and not in the interface, and thus the semantics is "neutral" w.r.t. the interface definition.

In this paper, after recalling the main concepts about agents and about web services, I will argue in favor of the usefulness of intelligent agents in the web services world.

2 Background: Agents

Agents are one of the important contributions of Artificial Intelligence about the nature of computing [3]. Agents are by definition software entities which interact with an environment, and are subject to modify themselves and evolve according to both external and internal stimuli, the latter due to the proactive and deliberative capabilities of agents themselves.

Agents are reactive in the sense that they can respond to stimuli (that can be, e.g., data coming form sensing devices, messages by other agents, or any kind of input). However, the type of response and the timing for the reaction to occur are in general by no means determined by a predefined pattern. In fact, agents are problem-solvers and

may have reasoning abilities: therefore, they react how and when they deem it appropriate. Reaction may imply performing actions to affect the environment. Agents are proactive in the sense that, according to past experience and internal reasoning, they are able take initiatives that may imply performing actions, but also setting an objective (or "goal") and constructing and executing a plan to achieve that goal. A "mentalistic" approach to agents [4, 5] considers objectives as "desires" and plans as "intentions". Agents' work is usually based upon a background knowledge base composed of "beliefs". Agents can be to some extent "intelligent", which from an observer's point of view means that agents are able to exhibit a flexible and adaptive behavior. As a pretty natural consequence of previously-mentioned features agents are autonomous, i.e., they are able to inhabit an environment and control their own behavior independently of external influence. Autonomous agents are typically stateful and persistent.

There are many approaches to agents in Computational Logic and Logic Programming (for a survey, the reader may refer to [6], [7] and to the references therein). Computational logic in fact can naturally represent agent's features (possibly by adopting modal logics) and semantic approaches have been devised to account for evolution and self-modification (cf., e.g., [8, 9]). Among the approaches to agents in Logic programming I mention (and often implicitly refer to) DALI ([10, 11, 8]), KGP ([12, 13]) EVOLP ([9, 14]) and for agents based on temporal logic I wish to mention the METATEM approach [18] [19].

As agent systems become more widely used in real-world applications, the issue of verification is increasingly important (see [7] and the references therein). In computational logic, two common approaches to the verification of computational systems are model checking [17] and theorem proving. There are many attempts to adapt these techniques to agents (see again [7]).

3 Background: Web Services

Service-oriented architectures (SOA's) are particular distributed systems whose agents are stateless entities communicating through standard network protocols and data formats. The Web itself can be seen as a SOA [20].

A particular instantiation of a SOA is the Web Service Architecture, whose agents, called web services, are "software systems designed to support inter-operable machine-to-machine interaction over a network. [...] They have an interface described in a machine-processable format [...] Other systems interact with the Web services in a manner prescribed by their description [...]".

Web services are, in practice, transient and stateless processes that exist only during service execution, which is triggered by a request coming from a consumer, or client. Services are instantiated to perform specific tasks, thus facilitating scalable, concurrent service provision. The design of a web service is usually defined as a clearly articulated workflow, for the sake of reliability and quality of service.

In recent years, web services technologies have demonstrated usefulness and promising capabilities in deploying complex, component-based, distributed applications operating in heterogeneous environments. Moreover, the standardization of the web services definition languages aims at making them inter-operable and platform independent. In

fact, in a SOA it is advantageous for service consumers and providers to obtain specific information on the services that they require or offer, respectively. The expression "semantic Web Services" [22–24], is commonly used to address such kind of web services, exposing an unambiguous and machine-processable description of their properties, interfaces, capabilities, and effects. Presently however, web services standards do not provide a means to build semantic web services, or lack a complete view by focusing on particular features, mostly security.

Semantic web services should support the development of several important technologies that are still lacking in the web services framework:

- automatic verification of web services behavior, as a design-time tool to ensure a correct implementation.
- automatic run-time verification of the web service adherence to its published specification.
- automatic discovery of the web services based on semantic information rather than on simple keyword-based matching.
- automatic generation of optimal (w.r.t. properties like cost, complexity, security etc.) service integrations ("choreographes").

The lack of these technologies is usually addressed as one of the main obstructions to the wide adoption of the web services architecture in the software industry.

The W3C Web Services Architecture working group has stressed the importance of attaching semantic meta-data to web services several years ago, concluding that the current specification formalisms are insufficient to this purpose. However, current standards are commonly considered to be not yet expressive enough to represent the semantic complexity required to describe a service and build effective tools for semantic verification, discovery and integration.

4 Agents and Web Services

Clearly, web services can be considered as purely reactive stateless agents with no problem-solving or proactivity capabilities. As Payne points out however, this extreme simplicity (aimed at efficiency, reliability and scalability) shows some limits in resource-bound environments, where the available processing power is limited or where services support physical equipment. Conflicts can occur when demand exceeds supply and this can lead to either failure or delay in service execution. Here, more powerful agents equipped with the capability of cooperating and forming commitments and contracts would clearly be of use.

I may add that, as it is easy to see, service composition can be modeled as an automatic planning problem where in the resulting plan (composed service) each service invocation corresponds to an action, modeled with inputs, outputs, preconditions, and effects. In association to the plan, suitable constraints on control, data flow, performance etc. should be introduced and dynamically verified. As "intelligence" is computationally costly, for reconciliating scalability and intelligence one way may be that of creating frameworks where various degrees of intelligence are distributed over various levels of the architecture. The resulting society might be composed of high-level intelligent

agents, aided in their tasks by (bunches of) elementary agents. The high-level agents are responsible of overall system strategies and plans, to be possibly devised in cooperation. (Bunches of) elementary agents may then come into play where either high efficiency or high parallelism or mobility is required.

Most presumably moreover, the next frontier of Artificial Intelligence will be the change of the perception of computers from "application-oriented" machines to "social" machines supporting large groups of people living and collaborating within an increasingly distributed network including also (in the Ambient Intelligence perspective) everyday-life appliances.

The very concept of SOA's is evolving toward "cloud computing". I mention from Wikipedia: "Cloud computing means Internet ('Cloud') based development and use of computer technology ('Computing'). It is a style of computing where IT-related capabilities are provided *as a service*, allowing users to access technology-enabled services without knowledge of, expertise with, or control over the technology infrastructure that supports them. It is a general concept that incorporates software as a service, Web 2.0 and other recent, well-known technology trends, where the common theme is reliance on the Internet for satisfying the computing needs of the users."

In this scenario, agents may come into play as "Intelligent Directories" that allow for advanced service retrieval. The "mentalistic" notions of desires and intentions of agents can be coupled to desires and intentions of users. Thus, Intelligent Directory agents should encompass not only the ability or reasoning about service definitions in Semantic Web terms (i.e., inference aimed at understanding whether the functional and non-functional specification of a service meets operational needs) but also in terms of user's expectations, preferences, fears, etc. They also might take into account previous experience, advice and/or feedback by others, etc. In a way, what is presently performed by humans in "social" web sites should be also performed by agents in next-generation SOA's.

In my opinion, important research trends in Computational Logic concern, just to start with:

- The investigation of how to enhance current frameworks and tools for the definition, composition and verification of web services. Particular attention should be devoted to the definition of both functional and non-functional properties in "human" terms rather than only in "machine-oriented" terms, and to their static and dynamic verification.
- The modeling of enhanced web services and of their combinations as reactive and proactive agents organized in multi-agent systems. This includes the modeling of service users as distinguished reactive and proactive agents that provide the system with their specific context-dependent needs, expectations, goals, preferences and constraints to be fulfilled. These systems should include social rules and roles, and agents should be able to learn in various ways, including learning "by being told" by trusted parties.

I believe that multi-layer agent frameworks (as an example I mention the one proposed in [15] and [16]) can be suitable for the above-mentioned aims: in fact, many relevant aspects can be modeled at the meta-level. On the one hand for instance, agents'

behavior can be customized and tailored to specific contexts. On the other hand, it is possible to define meta-axioms to be used at run-time for dynamically verifying the behavior and the properties of agents/services. These axioms can possibly be based on variants of temporal logics. Their verification may lead to undertake suitable "repair" or "improvement" actions that should restore/mantain expected/desiderable functionalities and performance. These actions may imply significant modifications to the modules adopted to implement services and to the plans adopted to implement choreographes.

5 Conclusions

Agents and web services have been abstractions on computation with clear relationships but also a clear separation under many respects. In this paper, I have argued that, in the view of the forthcoming "social" networks various concepts coming from the agents field might be profitably merged into the web service world. I have also argued that Computational Logic may have a role in this process.

References

1. Payne, T.R.: Web services from an agent perspective. IEEE Intelligent Systems **23**(2) (2008) 12–14 editorial.
2. Shi, X.: The challenge of semantic web services. IEEE Intelligent Systems **23**(2) (2008) 12–14 editorial.
3. Wooldridg, M.
4. Bratman, M.E.: Intention, Plans, and Practical Reasoning. Harvard University Press, Cambridge, MA (1987)
5. Rao, A.S., Georgeff, M.P.: BDI-agents: from theory to practice. In: Proceedings of the First Intl. Conference on Multiagent Systems, San Francisco (1995)
6. Torroni, P.: Computational logic in multi-agent systems: recent advances and future directions, special issue on computational logic in multi-agent systems. Annals of Mathematics and Artificial Intelligence **42**(1-3) (2004) 293–305 Invited contribution.
7. Fisher, M., Bordini, R.H., Hirsch, B., Torroni, P.: Computational logics and agents: a road map of current technologies and future trends. Computational Intelligence Journal **23**(1) (2007) 61–91
8. Costantini, S., Tocchio, A.: About declarative semantics of logic-based agent languages. In Baldoni, M., Torroni, P., eds.: Declarative Agent Languages and Technologies. LNAI 3229. Springer-Verlag, Berlin (2006)
9. Alferes, J.J., Brogi, A., Leite, J.A., Pereira, L.M.: Evolving logic programs. In: Logics in Artificial Intelligence, Proc. of the 8th Europ. Conf., JELIA 2002. LNAI 2424, Springer-Verlag, Berlin (2002) 50–61
10. Costantini, S., Tocchio, A.: A logic programming language for multi-agent systems. In: Logics in Artificial Intelligence, Proc. of the 8th Europ. Conf.,JELIA 2002. LNAI 2424, Springer-Verlag, Berlin (2002)
11. Costantini, S., Tocchio, A.: The DALI logic programming agent-oriented language. In: Logics in Artificial Intelligence, Proc. of the 9th European Conference, Jelia 2004. LNAI 3229, Springer-Verlag, Berlin (2004)

12. Bracciali, A., Demetriou, N., Endriss, U., Kakas, A., Lu, W., Mancarella, P., Sadri, F., Stathis, K., Terreni, G., Toni, F.: The KGP model of agency: Computational model and prototype implementation. In: Global Computing: IST/FET International Workshop, Revised Selected Papers. LNAI 3267. Springer-Verlag, Berlin (2005) 340–367
13. Kakas, A.C., Mancarella, P., Sadri, F., Stathis, K., Toni, F.: The KGP model of agency. In: Proc. ECAI-2004. (2004)
14. J.Alferes, J., Brogi, A., Leite, J.A., Pereira, L.M.: An evolvable rule-based e-mail agent. In: Procs. of the 11th Portuguese Intl.Conf. on Artificial Intelligence (EPIA'03). LNAI 2902, Springer-Verlag, Berlin (2003) 394–408
15. Costantini, S., Tocchio, A., Toni, F., Tsintza, P.: A multi-layered general agent model. In: AI*IA 2007: Artificial Intelligence and Human-Oriented Computing, 10th Congress of the Italian Association for Artificial Intelligence. LNCS 4733, Springer-Verlag, Berlin (2007)
16. Costantini, S., Acqua, P.D., Pereira, L.M.: A multi-layer framework for evolving and learning agents. In M. T. Cox, A.R., ed.: Proceedings of Metareasoning: Thinking about thinking workshop at AAAI 2008, Chicago, USA. (2008)
17. Clarke, E.M., Lerda, F.: Model checking: Software and beyond. Journal of Universal Computer Science **13**(5) (2007) 639–649
18. Barringer, H., Fisher, M., Gabbay, D., Gough, G., Owens, R.: MetateM: A framework for programming in temporal logic. In: Proceedings of REX Workshop on Stepwise Refinement of Distributed Systems: Models, Formalisms, Correctness. Volume 430 of Lecture Notes in Computer Science., Springer-Verlag (1989)
19. Fisher, M.: Metatem: The story so far. In Bordini, R.H., Dastani, M., Dix, J., Fallah-Seghrouchni, A.E., eds.: PROMAS. Volume 3862 of Lecture Notes in Computer Science., Springer (2005) 3–22
20. : Oasis soa reference model tc http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.
21. Al-Masri, E., Mahmoud, Q.H.: Discovering the best web service. In: WWW '07: Proceedings of the 16th international conference on World Wide Web, ACM Press (2007)
22. : W3c semantic web activity http://www.w3.org/2001/sw/.
23. : Owl-based web service ontology http://www.daml.org/services/owl-s/.
24. S. A. McIlraith, T.C.S., Zeng, H.: Semantic web services. IEEE Intelligent Systems **16**(2) (2001) 46–53