# What Can Agent-Based Computing Offer Service-Oriented Architectures, and Vice Versa?

Wayne Wobcke[1], Nirmit Desai[2], Frank Dignum[3], Aditya Ghose[4], Srinivas Padmanabhuni[5], and Biplav Srivastava[6]

[1] School of Comp. Sci. and Eng., University of New South Wales, Sydney NSW 2052, Australia
wobcke@cse.unsw.edu.au
[2] IBM India Research Lab, Embassy Golf Links Business Park Bangalore 560071, India
nirmit.desai@in.ibm.com
[3] Dept of Information and Comp. Sciences, Utrecht University, 3508 TB Utrecht, The Netherlands
dignum@cs.uu.nl
[4] School of Comp. Sci. and Software Eng., University of Wollongong, Wollongong NSW 2522, Australia
aditya@uow.edu.au
[5] Software Engineering and Tech. Labs, Infosys Technologies Ltd, Bangalore 560100 India
srinivas_p@infosys.com
[6] IBM India Research Lab, Block 1, IIT Campus, Hauz Khas, New Delhi 110016, India
sbiplav@in.ibm.com

**Abstract.** This article serves as a record of a panel discussion held at PRIMA in November, 2010. The panel consisted of two academic and three industry representatives, and thus provided a rare opportunity to discuss the relationship between agent-based computing and service-oriented architectures from both points of view. The basic question for the panel was to identify the key research and industry issues that arise in the deployment of systems based on service-oriented architectures, and in particular to address whether the agent-based computing paradigm offers any resolution of those issues. The question was also posed whether applications based on service-oriented architectures provide a suitable platform for implementing agent-based systems, which are presently limited in application by comparison. This summary is presented with the aim of stimulating further academic and industry collaborative research in this fast growing area which potentially has wide-ranging practical application.

## 1 Introduction: Wayne Wobcke (University of New South Wales)

By now it is hardly news that there is a close relationship between service-oriented architectures [8] and agent-based computing. It was noticed quite early that, at a technical level, service-oriented computing platforms would require mechanisms for service discovery (of the sort used in agent-based platforms such as KQML [6]), service aggregation or composition (analogous to planning complex series of actions) [13], coordination of multiple services (similar to multi-agent plan coordination [7]), execution monitoring (as agent systems monitor plan execution), and quality assurance (involving mechanisms for selection of appropriate actions in dynamic environments, a central concern of rational agent architectures [11]).

N. Desai, A. Liu, and M. Winikoff (Eds.): PRIMA 2010, LNAI 7057, pp. 1–10, 2012.
© Springer-Verlag Berlin Heidelberg 2012

The purpose of this panel discussion is to reconsider this connection in the light of nearly a decade of industry experience during which service-oriented architectures (SOA) have become mainstream in the software industry. The basic question is whether the close technical relationship to agent-based computing still exists, and if so, whether this connection is of purely theoretical interest or has genuine practical implications, and, further, whether there are more fundamental obstacles to the deployment of SOA-based systems in industry contexts that are not covered by the narrow technical view outlined above. The implicit objective is to formulate a research agenda for the short-to-medium term that would enable agent researchers to contribute to the fast growing area of service-oriented computing, and (this is the "vice versa" part) to consider whether service-oriented architectures offer suitable platformis for the implementation and ultimately commercialization of agent-based systems. A concern here is whether there is currently the degree of flexibility and interoperability in commercial service-oriented computing platforms required to support agent-based applications. To this end, we are grateful for the participation on the panel of representatives from Infosys Technologies and IBM Research India who are able to provide an industry perspective on these issues.

The topic of the panel is deliberately framed towards technical aspects to encourage concrete discussion, in doing so presupposing that this close underlying technical connection exists, and this partly to provoke panellists into possibly rejecting this assumption (none of them did). Of course it is also recognized that "services" are far broader than just SOA-based systems, and so the panellists are also invited to comment on wider-ranging issues. Though not specifically the topic of the panel, the question of the role of standardization generally and of standard ontologies and their associated reasoning frameworks is one that naturally arises during discussion.

In keeping with the open ended nature of panel discussions, there is no formal conclusion given here. Readers must draw their own conclusions based on the panellist statements that follow.

## 2 Frank Dignum (Utrecht University)

### 2.1 ALIVE: The Role of Agents in Adaptive Service-Oriented Architectures

Web services [5] and service-oriented architectures [9] have the potential to increase significantly the utilization, compatibility and interoperability of information and communication systems. This progress has, for the first time, raised the realistic possibility of deploying large numbers of services in companies' and public organizations' intranets and extranets, and in the public Internet, in order to create communities of services which are always connected, always changing, open or semi-open, and form the baseline environment for software applications. However, this shift brought about not only potential benefits, but also serious challenges about how such systems and applications should be designed, managed and deployed. Existing approaches in some important areas (such as security, transactions and federation) tend to only cover technology issues such as, for example, how to secure a protocol or connect federated directories, without considering the paradigm change that occurs when large numbers of services are deployed and managed over time. In particular, existing approaches do not offer satisfactory solutions to the following issues:

- How to dynamically compose services into workflows that serve a specific purpose and adhere to some overall requirements (like efficiency, security, etc.).
- How to align the configurations and settings, needed by a service to operate, to the operational environment.
- How service execution is affected by issues of trust, rights, obligations and permission.
- What if critical applications simply cease to function if services provisioned from third parties disappear or malfunction?
- How to deal with knowledge representation, when connecting or binding together two or more actual entities or services using different ontologies.

All these issues point to the need for a "social layer" as part of the service interaction context. From an engineering perspective, new approaches are needed which take a holistic view of service environments, and take into account not only the properties of individual applications, but also the objectives, structure and dynamics of the system as a whole. In the ALIVE project[1] [4,15], we have combined existing work in coordination and organizational models with the state-of-the-art in service-oriented computing. The project extends current trends in service-oriented engineering by adding three extra layers [3]:

- The *organization layer* provides context for the other levels, specifying the organizational roles, objectives and rules that govern interaction and using developments in organization dynamics to allow structural adaptation of distributed systems over time.
- The *coordination layer* provides the means to specify, at a high level, the patterns of interactions between services, using a variety of coordination techniques. At this level agent technology is used.
- The *service layer* augments the existing service models with semantic descriptions to make components aware of their social context and rules of engagement with other services.

In practical terms, agent solutions combined with organization structures facilitate the implementation of purpose-oriented workflow mechanisms. The organization model defines the purpose of the content composition – e.g. metrics for quality of information, interaction patterns, acceptable processing time, etc. The workflow actors inherit the goals and plan rules to implement these characteristics from the organization structure. Using the three layers we are able to divide the knowledge and abilities that are necessary to dynamically create and maintain the complex workflows of services in a natural way.

At the service layer we concentrate on the knowledge necessary to see which service is best suited for a certain purpose. For example, there might be many weather prediction services; some predict weather for only one day, others for a week. If we want to plan a holiday, we want to check the weather for a whole period but it does not have to be very accurate, whereas for a farmer accuracy might be very important.

[1] ALIVE Project: http://www.ist-alive.eu/

At the coordination layer we use typical agent methods to create plans of service invocations to reach certain goals. Agents can interact to combine their plans in order to profit from each other's services. Especially useful is the fact that agents can recover from failures of their plans and replan for a goal. This is very difficult to do in current service-oriented tools.

At the organization level the overall objectives of the system can be specified such that the autonomy of the agents is only used in order to reach those objectives. This is also the level where service level agreements can be specified that should be fulfilled by the service(-compositions) and the agents.

In the ALIVE project we have tested the above sketched framework in several use cases. Although we can conclude that the framework indeed is useful, it also needs some perseverance to get things working. Compared to a more traditional service-oriented approach a lot more constructs have to be specified and implemented. We need to specify semantics for the services, tasks and plan rules for the agents, interaction patterns for the agents to create workflows, organizational structures in which the agents have to function, etc. These structures only start paying off with complex systems, especially when services change, fail or are aborted. In these cases the ALIVE framework provides a very high level of robustness.

A second lesson learnt is that it is far easier to construct the whole framework for a specific application than to generate a software engineering tool set that can create the framework for many different applications. We used a model driven approach to connect the elements of the different layers. This does help to keep consistency throughout the framework. However, it also means that meta-models have to be available for every module in the framework. This is not trivial if existing (and especially third party) components are used, e.g. one needs a meta-model for the agents (including their plan structures).

Also in the ALIVE framework we had to device a general way for agents to find the most suitable service to execute a (part of a) plan. Because the steps in a plan are usually not all instantiated before executing the plan (in order to allow for flexibility in planning) the queries for services also contain variables which have to be dealt with on the service level and possibly passed back up. These mechanisms are not part of traditional service-oriented methods and take much work and care to construct. This is not just the case for the way we implemented the ALIVE framework, but is inherent in any agent driven service-oriented system. Once agents are used to flexibly use and compose the services, one needs this type of query mechanism that can deal with requests for services that are not (fully) instantiated. Thus there is a seemingly inherent trade-off between efficiency of implementing (note: not efficiency of the implementation) an application and the flexibility of the system. The need of the flexibility of the system should warrant the extra effort in the specification and design of the system. In our use cases, flexibility was needed because services could fail and be changed on the fly. An extreme case is that of our crisis management use case [10] where services might fail at any moment due to the crisis, but you need your system to handle the crisis properly nonetheless.

# 3 Biplav Srivastava (IBM Research – India)

## 3.1 The Problem Context

Changes are continuously happening in enterprises and they impact the Information Technology (IT) landscape. This leads to widespread needs like quickly delivering new applications and integrating existing applications. However, application development is often done in an ad-hoc manner resulting in poor reuse of software assets and longer time-to-delivery. Service-oriented architectures like web services have received much interest due to their potential in facilitating seamless business-to-business or enterprise application integration. A web service composition system can help automate the process, from specifying business process functionalities, to developing executable workflows that capture non-functional (e.g. QoS) requirements, to deploying them on a runtime infrastructure. Intuitively, web services can be viewed as software components and the process of web service composition similar to software synthesis. In addition, service composition needs to address the buildtime and runtime issues of the integrated application, thereby making it a more challenging and practical problem than software synthesis.

## 3.2 The Case for Service-Oriented Architecture and Issues Learnt in the Field

There are many approaches for composing and executing web services (see the survey [2]) and open problems [13]. Synthy is an example of one of the approaches which has been tried in the enterprise setting [1]. It is based on a novel two-staged composition approach that addresses the information modelling aspects of web services, provides support for contextual information while composing services, employs efficient decoupling of functional and non-functional requirements, and leads to improved scalability and failure handling. Synthy is a technology for semi-automatically composing SOA-compliant components such that the new component meets the desired functional and non-functional requirements and the resultant component can be flexibly executed.

The experience from the field has been that SOA can indeed help with application integration [14]. But there are many issues in practice:

- Domain modelling is hard.
  - SOA needs modelling of services, and if Business Process Management (BPM) is being followed, the business services. But this is not easy. A common problem is which domain expert to believe.
  - Companies in monopolistic situations (e.g. Microsoft, SAP) have an easier time.
  - Domain experts are expensive and there is an open question on the quality of models built by typical IT professionals.
  - An open research issue is to determine the right level of abstraction.
- Managing runtime is hard.
  - How to prove a composition of services is correct at runtime? There is a human-in-the-loop requirement for many applications.
  - Graceful degradation during runtime is often required.
- Interoperable tooling is unavailable.

### 3.3 The Case for Agent-Oriented Computing in SOA

Agent-oriented computing has delved extensively in modelling and coordination issues for autonomous agents. Moreover, the community has experience in designing, simulating and executing agent-based solutions to long-running, mission-critical defence problems. These are exactly the areas where SOA needs help.

The agent community needs help in standardization and wider adoption by mainstream business. SOA has lessons on how to make a technology widely usable. After all, WSDL, UDDI and BEPL4WS are the mainstay of modern SOA IT platforms, and very widely supported by major IT vendors.

## 4 Srinivas Padmanabhuni (Infosys Technologies)

### 4.1 Agent Orientation: Complementing Process and Service Orientation for Ultimate Flexibility

With the increase in the complexity of IT systems, it has become difficult for administrators to manually maintain and tune IT systems to meet the requirements of individual consumers. To meet the increasing complexity of IT systems, there is a requirement for the systems to become more human independent and self-managing. We firmly believe agent technologies are a potent technology to address the issue of IT complexity, especially when viewed from the lens of flexible business processes.

As already well established, a service-oriented foundation, forming the bridge between IT implementations and business processes, is at the centre of the future proof enterprise process architectures. Hence, service-oriented architecture (SOA) is considered an inherent foundational base for today's Business Process Management (BPM) implementations, wherein individual services form the crucial business activities, and orchestration of the services forms the basis of executable business processes.

However, flexibility at process level is incomplete without a thorough understanding of the different variations possible in the manifestations of an individual service as part of a dynamic business process. In this context of dynamic reconfigurability of individual services, we envisage a crucial role agent-based systems can play. The issue of dynamic reconfigurability of individual service implementations in dynamic processes is an important problem. Their need is to reconfigure themselves and coordinate with participating components automatically (without human interaction) to cater to the changing consumer requirements. We have researched the role of an agent-based approach to endow service variations, with the ability to dynamically reconfigure services automatically to meet the needs of their users. The role of agent-based architectures lies in dynamically sensing in an autonomic mode the external environmental variables, and thereupon dynamically evolve the corresponding service implementation by embodying the right variation, to evolve the right service interface, which will be the final true face of the service, in the ongoing dynamic business process. Our ongoing research is looking at a systematic exploration of several combinations of multi-agent system tools and protocols in conjunction with dynamic services-based business processes.

Yet another area where agents are relevant is in the problem of policy reconciliation of multiple actors interacting via service interfaces. We have researched approaches using soft constraints to provide an extensible and flexible mechanism for reconciliation

of policies between multiple interacting actors. In the context of multiple actors needing to collaborate together to carry on shared activity, or a sequential activity with dependence upon one another, the policy constraints need to be mutually consistent in order to carry on a transaction. Especially in context of business-to-business (B2B) business processes, where two heterogeneous actors work together as part of a B2B process, this kind of policy reconciliation is a must. We are researching the role agent systems can play in effective and dynamic policy reconciliation for B2B processes. The framework is applicable to any heterogeneous environment in need of reconciling policies, and is illustrated in the real business use case of a demand driven supply chain framework. For details of the preliminary work in this area, please see [12].

Overall, we see promise for multi-agent systems technology working to enable a truly autonomic and dynamic environment for flexible service-based executable business processes.

## 5 Nirmit Desai (IBM Research – India)

### 5.1 Services Industry is the Application Area "Agents" Have Been Waiting For!

The agent research community can benefit immensely by demonstrating what their research can do for the services industry. Before I get into backing that statement up, let me say what "services" are and what they are not.

- Services cannot be supported by SOA as it is. When services are proclaimed as having major economic significance, we are not referring to services as in SOA. We are referring to business services. SOA is too low-level a concept to support services.
- Services are not invoked, they are engaged. When was the last time you "invoked" your domestic help? How about the health care service? If you were to believe SOA, these services would have a WSDL description. They would take input from you, go off somewhere, and come back with a cleaned house or a mended tooth.
- Services are hardly automated. Would you go to a robot to have your disease diagnosed? It is good if the robot can serve you but we are far from it. As a result, services involve people. There are specialized skills and deep domain knowledge in almost any services industry. There is an aspect of face-to-face interaction.
- Services are measured by satisfaction. The customer needs vary greatly and they evolve with time. Nonetheless they need to be satisfied to "buy" a service again. However, "satisfaction" is not well understood. Can the customer be satisfied if the provider meets the service level agreements (SLAs)? Can the customer be satisfied even though some SLAs have been violated?

Most importantly, services comprise a major part of the world's economy. Unfortunately, most of the work by computer scientists that is branded as "services" research does not meet these criteria. We need to meet these criteria because they characterize services in a true sense. Fortunately, the multi-agent systems community has worked on the principles underlying services for decades. So why worry about services now? And why should we as an agent research community care?

Distributed AI as an area has for long taken on this apparently difficult mission. We have theories that explain how agents ought to communicate in a business environment and how they can fulfil the needs of their principal. We have agents who are smart enough to interact with humans. We draw ideas from philosophy, social sciences, and cognitive psychology. We study automated negotiation and argumentation. We study trust and commitments. We study rationality and decision making. All of these belong to the heart of the service science.

So what is missing? Game changing applications. The Distributed AI community cannot boast of scientific impact that several other fields of Computer Science can do. For example, communication networks have revolutionized how we communicate, relational databases have revolutionized business transactions. There is a need to justify our programs of research. So far, services have not been a favourite application area of scientists for three main reasons: (1) it is a low-margin and cost-based business – to the service providers, immediate solutions have infinitely more "perceived" value than a long-term scientific effort, (2) the fundamental issues in this area are too hard to make an impact on, and (3) services did not command major economic significance.

However, there are two encouraging trends: (1) services have grown to be the largest chunk of the world's economy, and (2) we are starting to see a certain degree of success in attacking these difficult problems. For example, we have *Watson* that can play Jeopardy and beat the best players ever to play that game. While *Deep Blue* was 14 years ago, chess is not exactly an area of difficulty to computers. Still, what *Watson* and *Deep Blue* have accomplished is far short of the holistic vision of multi-agent research. This is why we need to care for applying our research to the services industry.

## 6 Aditya Ghose (University of Wollongong)

### 6.1 An Agent-Based Response to the Climate Change Challenge[2]

The climate change crisis presents both a challenge and an opportunity of unprecedented proportions to the agent community. Current thinking on climate change responses emphasizes the development of alternative energy sources, the development of smart automotive technology and the introduction of macro-economic levers (e.g. carbon taxes, emission trading schemes etc.) to alter energy consumption behaviour at the level of both enterprises and individuals. Fundamental to any solution to the problem is *efficient planning and optimization* (in particular, ensuring that energy use is optimized) – yet this has been largely ignored in the current discourse.

Reducing energy consumption requires that we seek to make all behaviour efficient, *everywhere, all the time*. This requires *pervasive, distributed, continual, reactive* and *autonomous* decision support. The agent community prides itself on its ability to deliver systems with precisely these attributes.

The *Optimizing Web* project at the University of Wollongong offers an example of what can be achieved. The project is based on the following observations. First, optimization is fundamental to carbon mitigation – optimization enables efficient resource

[2] This response was prompted by a question on the "grand challenges" in a future "services science".

utilization, thus lowering energy consumption and the carbon footprint. Second, the global industrial/technological infrastructure, including transportation systems, manufacturing plants, human habitat and so on, is typically operated in an ad-hoc and significantly sub-optimal fashion. This remains the case despite the availability of sophisticated optimization technology for almost the past seven decades (present day operations research techniques trace their roots to the work of George Dantzig in the early 1940s that resulted in the original optimization algorithm – linear programming). Third, locally optimal behaviour does not guarantee "globally" optimal behaviour (i.e. if all agents in a multi-agent system adopt locally efficient behaviours, that does not guarantee that the behaviour of the system as a whole is efficient). Conversely, an optimal solution for a multi-agent problem might not necessarily be optimal for each of its constituent sub-problems. This suggests that more widespread uptake of "piecemeal" optimization alone will not work what is needed is a network of local optimizers that collaborate (and potentially negotiate) to obtain system-wide solutions that improve efficiency despite the competing pulls of local objectives.

The Optimizing Web leverages the global (near-)consensus (without being too pessimistic!) on a carbon-footprint minimization objective. It achieves large-scale *collaborative optimization*, where large numbers of agents collaborate to obtain an optimal value for a shared objective function. The vision is to provide ubiquitous collaborative optimization services, at the level of individual devices, vehicles within transportation systems, units within organizations or manufacturing plants – as well aggregations of all of these. The optimizing web provides a set of protocols for local optimizing agents to interoperate to improve the value of a global carbon footprint minimization objective, while making appropriate trade-offs in relation to their local objectives.

The Optimizing Web leverages and integrates two aspects of agent technology: (1) distributed constraint optimization (DCOP) and (2) distributed reactive planning. While we know that planning problems can be formulated as optimization problems, it is also well understood that some problems are more naturally modelled as planning problems, while others as optimization problems. The project therefore leverages DCOP insights for distributed optimal reactive planning.

Ultimately, the agent community needs to do much more along similar lines. The climate change crisis is real, and the agent community has real solutions to offer. This is therefore a call to arms.

**Acknowledgements.** We would like to thank Smart Services Cooperative Research Centre for its support of PRIMA 2010 which provided the impetus for this panel discussion.

## References

1. Agarwal, V., Chafle, G., Dasgupta, K., Karnik, N., Kumar, A., Mittal, S., Srivastava, B.: Synthy: A System for End to End Composition of Web Services. Journal of Web Semantics 3, 311–339 (2005)
2. Agarwal, V., Chafle, G., Mittal, S., Srivastava, B.: Understanding Approaches for Web Service Composition and Execution. In: Proceedings of the 1st Bangalore Annual Compute Conference (2008)

3. Aldewereld, H., Penserini, L., Dignum, F., Dignum, V.: Regulating Organizations: The ALIVE Approach. In: Proceedings of the International Workshop on Regulations Modelling and Deployment (ReMoD 2008) Held in Conjunction with the CAiSE 2008 Conference, pp. 37–48 (2008)
4. Álvarez-Napagao, S., Cliffe, O., Vázquez-Salceda, J., Padget, J.: Norms, Organisations and Semantic Web Services: The ALIVE Approach. In: Proceedings of the Workshop on Coordination, Organization, Institutions and Norms in Agent Systems in Online Communities at MALLOW 2009 (2009)
5. Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchad, D.: Web Services Architecture. W3C Working Group Note 11, The World Wide Web Consortium (W3C) (February 2004)
6. Finin, T., Fritzson, R., McKay, D., McEntire, R.: KQML as an Agent Communication Language. In: Proceedings of the Third International Conference on Information and Knowledge Management (CIKM 1994), pp. 456–463 (1994)
7. Jennings, N.R.: Controlling Cooperative Problem Solving in Industrial Multi-Agent Systems Using Joint Intentions. Artificial Intelligence 75, 195–240 (1995)
8. Papazoglou, M.P., Georgakopoulos, D.: Service-Oriented Computing. Communications of the ACM 40, 25–28 (2003)
9. Papazoglou, M.P., van den Heuvel, W.-J.: Service Oriented Architectures: Approaches, Technologies and Research Issues. The VLDB Journal 16, 389–415 (2007)
10. Quillinan, T., Brazier, F., Aldewereld, H., Dignum, F., Dignum, M.V., Penserini, L., Wijngaards, N.: Developing Agent-Based Organizational Models for Crisis Management. In: Proceedings of the Industry Track of the 8th International Joint Conference on Autonomous Agents and Multi-Agent Systems (2009)
11. Rao, A.S., Georgeff, M.P.: BDI Agents: From Theory to Practice. In: Proceedings of the First International Conference on Multi-Agent Systems (ICMAS 1995), pp. 312–319 (1995)
12. Schmid, A., Padmanabhuni, S., Schroeder, A.: A Soft Constraints-Based Approach for Reconciliation of Non-Functional Requirements in Web Services-Based Multi-Agent Systems. In: Proceedings of the 2007 IEEE International Conference on Web Services, pp. 711–718 (2007)
13. Srivastava, B., Koehler, J.: Web Service Composition - Current Solutions and Open Problems. In: Proceedings of the ICAPS 2003 Workshop on Planning and Scheduling for Web Services (2003)
14. Srivastava, B., Mazzoleni, P.: Business Driven Consolidation of SOA Implementations. In: Proceedings of the 2010 IEEE International Conference on Services Computing, pp. 49–56 (2010)
15. Vázquez-Salceda, J., Dignum, F., Vasconcelos, W., Padget, J., Clarke, S., Ceccaroni, L., Nieuwenhuis, K., Sergean, P.: ALIVE: Combining Organizational and Coordination Theory with Model Driven Approaches to Develop Dynamic, Flexible Distributed Business Systems. In: Telesca, L., Stanoevska-Slabeva, K., Rakocevic, V. (eds.) Digital Business. Springer, Berlin (2009)