

Panel Report:

**Is it *feasible* to build *effective* multi-vendor
autonomic computing systems?**

**at 3rd IEEE International Conference
on Autonomic Computing (ICAC 2006)**

Omer F. Rana Cardiff School of Computer Science & Welsh eScience Center, UK o.f.rana@cs.cardiff.ac.uk	Jeffrey O. Kephart IBM T.J.Watson Research Center, USA kephart@us.ibm.com
--	---

The panelists were:

- Steve White (SW), IBM Research, USA
- Jose Fortes (JF), University of Florida, USA
- Kumar Goswami (KG), HP Labs, USA
- Julie McCann (JM), Imperial College, London, UK
- Mazin Yousif (MY), Intel Research, USA

The panelists were asked the following questions:

1. What does an “autonomic computing” system mean to you and your organisation/research? How do you sell this idea to your customers or immediate research community.
2. What are the *key* technologies needed to establish multi-vendor autonomic computing systems? How far away are we from realizing these technologies—or do these already exist?
3. What are the *key* barriers (political/economic/social) that may prevent the use of autonomic computing systems in real world business and/or scientific applications? How do you think can these be overcome?

The panelists provided a variety of viewpoints on these topics from both an industry and an academic perspective. SW suggested that existing enterprise IT systems were already multi-vendor, especially if one considers both desktop and server based applications. Furthermore, there was a growing trend towards multi-vendor outsourcing, whereby different companies providing outsourcing support were required to handle independently: networking, databases, server, storage, etc. According to SW, many of these individual subsystems had requirements for autonomic behaviour. However, present-day systems fell far short of

being autonomic: the constituent subsystems were not interoperable with one another across siloes, tiers, and vendors. SW indicated that it would therefore be necessary to deploy autonomic system ideas in the context of such an existing IT “ecosystem”. Addressing the issue of “complexity” associated with managing such system components remained the most significant motivation for using an autonomic system. Doing so will entail (among other things) extensions to WSDM to endow self-managing resources with QoS-based interfaces, and a focus on both automating and eliminating the steps of end-to-end IT processes.

This vision was also shared by the other panelists. JM indicated that an autonomic system should not just be the development of a static resource manager/optimiser. Instead it should take a much wider perspective, attempting to define various aspects of an autonomic computing (or self-adapting system) architecture. JM indicated that interoperability would be easier to achieve if such an architecture decoupled autonomic components from the functional components. KG agreed, advocating a service-oriented architecture with self-describing services as the basic unit of modularity. In his view, formal models, complete with a description of inter-module dependencies, are an essential element of autonomic systems. MY added that descriptions are also needed for other aspects of the environment, including the business, workloads, workflows, problem descriptions, etc. Also key to the success of autonomic computing, in KG’s view, is development of the full lifecycle of real-time monitoring, analytics that process the raw monitored data into higher-level descriptions of system state, and automated algorithms that determine and execute appropriate actions as needed, with machine learning techniques playing an important role in the analytic and decision-making phases.

Business benefits associated with developing autonomic systems were also emphasized by all panelists, such as streamlining data center management to increase the number of servers maintained by a single system administrator. Such business need was stressed by both MY and KG, suggesting that autonomic computing should aim to avoid the need for performing repetitive and boring tasks required of system administrators, instead allowing them to focus on the more “cognitive” skills. However, for such autonomic capability to be effectively deployed in real systems, KG indicated the importance of “trust” issues in automatic system administration. KG identified the “people equation” that could act as a barrier to the better take up of autonomic systems in real environments—such as the taking away of control from a system administrator, the ability to automatically manage interactions between components and being able to trust the system to automatically self-adapt. Fear of having one’s job replaced by automation is another factor that can lead administrators to eschew automation, no matter how effective it may prove to be in practice. Managing user expectations remained an important objective to enable wider adoption of autonomic computing systems. JM discussed another aspect of trust: it’s not just people that need to trust autonomic systems, but autonomic services need to trust (and *a fortiori* understand) one another’s capabilities. They need moreover to tolerate fuzziness across different semantic mappings, as there is unlikely to be any one *lingua franca*. Coping with such fuzziness (and resolv-

ing disagreements between what is requested and what is feasible) will require negotiation between service requesters and service providers.

MY focused on the value of autonomic computing for data centers, indicating that this approach was already an important component of “dynamic” data centers currently being planned. One of the key reasons for this takeup was the recognition that autonomic computing would provide an improved ability to managing Service Level Agreements (SLAs) and lead to administrative simplicity in the existing system. In this context, the need for developing suitable “policies” was also discussed by all the panelist. Especially the requirement to better synchronize business and IT metrics that are part of such a policy.

All of the panelists recognized the importance of standards to support autonomic computing. It was also recognized that standards could be considered at different levels of the software stack: Web Services standards (such as WSDM and WS-Transfer), standards for defining policies (WS-Policy, for instance), and standards for capturing resource information (CIM and JSDL). All of these standards play an important part in supporting multi-vendor interaction between autonomic components. SW emphasized that, to achieve self-managing resources, it was necessary to extend existing standards (such as WSDM, and not attempt to create new ones). JM proposed a more wider perspective, identifying possible areas where standardization was necessary, namely: the types of probes needed for monitoring (what data, when, and how often), the types of events generated within the system (which ones and when), and actions performed by the system (how would these be defined and what would they be). All of the panelists also identified the need for “tools” to allow application and component developers to define autonomic capability.

However, MY commented that the Web Services stack now had a very large number of specifications—and it was important to consider precisely which standards were relevant for implementing multi-vendor autonomic systems.

The importance of “virtualization” was also recognized by all of the panelists. Virtualization could be viewed from two perspectives: (i) as a means to aggregate computational and data resources and enable multiple viewpoints on these resources to co-exist; (ii) as an abstraction for a more uniform access mechanism to resources. In (ii), the intention would be to allow multiple vendors to share the same abstraction, but provide different types (perhaps with different specialization) of implementation of these abstractions. JF emphasized that such virtualization could benefit from work already being undertaken in this area within the Grid community.

MY identified some of the key barriers to deployment of autonomic systems as being interoperability between systems. For instance, information models and schemas for data exchange between autonomic system components, from different vendors, still could not be exchanged in a seamless manner. This was identified as the “semantic stack” – essentially the sets of information models that could exist at different levels of an autonomic system: ranging from low level components such as routers and servers, to intermediate level operating environments, to information associated with applications. JM indicated that this was a difficult challenge, as it remained an important requirement of a

number of other computing communities.

A lively Q&A session followed the panelists' statements. The panelists agreed that we need to attack problems from both a short-term and a long-term perspective. There still remains a lot of *low-hanging* fruit—there are many obvious things that we can automate in the near-term. One particular customer requirement that is still of importance today (compared to high reliability and performance) is to make components easier to manage. We also have to think today about the long term, and how to solve issues in a more holistic fashion than we tend to do when solving problems tactically.

Karsten Schwan challenged the panelists to define what they meant by “end-to-end”, as this term was used repeatedly by all. SW defined it in terms of the full lifecycle of IT processes such as change management. MY agreed, giving further examples such as the problem management lifecycle. JM and JF pointed out that there are many valid perspectives, as there are many different system users, each with their own role. This topic engendered some further discussion about the need for an application scenario that would demonstrate the effectiveness of autonomic computing ideas. Data Centers and Grid Computing applications provided useful candidates for such a scenario.

Summary

Overall, the panelists *concurred* on the following ideas:

- Autonomic systems were already in place—to a limited extent, and mainly at the level of subsystems such as networks, databases, storage systems, and servers rather than at the level of whole systems. To achieve wider adoption by customers and researchers, it was necessary to identify what “autonomic computing” would mean to such individuals. Such an undertaking was also necessary to build better “trust” in autonomic systems.
- Interoperability and virtualization were key requirements to build effective multi-vendor autonomic systems. Interoperability was required at different levels, and the various approaches to virtualization needed to be reconciled.
- Following infrastructure standards, and extending existing standards (rather than creating new ones), would also lead to further takeup of multi-vendor autonomic systems.
- It was necessary to identify key application scenarios that would enable wider takeup of multi-vendor autonomic systems. Data Centers and Grid computing applications provided useful exemplars to consider.