Lotus. Developer Domain | The Technical Resource for Lotus software | LDD Today

Autonomic Computing: It's about making smarter systems

Level: All
Works with: Lotus Workplace Messaging, Notes/Domino, Sametime
Updated: 02-Jun-2003

Interview by
Tara Hall

Autonomic computing? Just another industry buzz word or a true technology concept? This month we talk with Vaughn Rokosz, the Lotus software engineer heading up Quality Practices at Lotus, about autonomic computing, the efforts being made in IBM and Lotus, and how autonomic computing will change the industry.

What is autonomic computing?
Autonomic computing is really about making systems self-managing. This is a term that was coined by Paul Horn of **IBM Research** a year and a half or two years ago to help direct our attention away from our traditional notions of how we think about computer systems and more towards biological systems. If you think about biological systems like the human body, they're tremendously complex and very robust. The human body, for example, is constantly making adjustments. Your heart rate is being controlled; your breathing rate is controlled. All of these things happen beneath the level of conscious control. So biological systems give us a metaphor for thinking about computer systems. When we take a look at the attributes of biological systems, we can find attributes that we wish our computer systems had, like self-healing, self-configuring, and self-protecting. We can begin to build the attributes that we see in biological systems into complex computer systems. In the end, it translates into real customer benefits because these more complex systems are easier to administer.

What is your role in the IBM/Lotus autonomic computing effort that's currently going on?
Autonomic computing is a pretty big technology initiative right now across IBM. There are a lot of people involved from all sides: research and the brands. My role is to be one of the point people for the Lotus brand. I represent Lotus to the wider autonomic computing community at IBM. There's a lot of development going on inside IBM—technologies that we need to be aware of at Lotus so we can build systems that play in the larger autonomic systems. So one of my jobs is to bring that information back into Lotus.

What are the current problems within the industry that autonomic computing will solve?
It's really about complexity right now. As computing power has increased, we've got the ability to create much larger kinds of applications. With millions or tens of millions of computer systems all cooperating, this complexity comes at a cost because humans are sitting behind the scenes, making all these machines work together. What we hope will happen is to see autonomic computing behavior in the computer systems, so it becomes less costly for people to build these complex applications, and that, in turn, enables a whole new class of application to be developed.

Is the idea to make upgrading of a system or repairing a system totally hands off to the administrator?
Not necessarily. When we talk about self-configuring—this is really hitting on the upgrade issue—it would certainly be nice if it didn't require extensive analysis and that your systems could detect, for example, that they weren't at the correct release and automatically downloaded and installed the latest release. That's not necessarily what people want to do. Some people may actually prefer to have some control. Some of the research teams have put out the idea that autonomic computing isn't about making people go away; it's really changing the nature of the partnership between system administrators and the computers. It's putting more of the burden on the computers and less on the system administrators. That doesn't mean that the system administrator doesn't play a role.

Can you go into more detail about the benefits that this technology offers to administrators and to users?
In some ways, we hope it's invisible to users. In a sense, users really just want their systems to work. If the autonomic systems are successful, they will just work. It may be visible in one way—it reduces the number of times you have to call your help desk.

For administrators, we think it translates into the need to spend less time micromanaging their machines and more time thinking about what we believe are the real issues of what's going on in the business and what kinds of business policies are in place. So it really shifts where they have to focus. Today there's a focus on the minutiae of configuration and how to tweak this server parameter to get the best performance. If the machines were self-tuning, you wouldn't have to do that. You could take yourself up a level to think about more interesting issues, such as how much benefit to the company this section of my infrastructure is delivering to me.

Are there any benefits to application developers at this point or any intended benefits for them?
Yes, we think so. Some of the technologies that are even downloadable now directly affect application development. For example, you can download the **Log and Trace Preview for Autonomic Computing** in Beta form from IBM alphaWorks. The developer can then write J2EE applications and log in a standard format, then use more advanced tools to analyze log events across a large number of servers for better problem determination.

So far, what steps has IBM, and in particular Lotus, taken towards autonomic computing?
If we look back at Domino's history, we see a lot of autonomic behavior in Domino before the term autonomic computing came about. For example, clustering for failover and load balancing exemplifies autonomic behavior: self-optimizing and self-healing technologies. If I have a Domino cluster and one of the machines fails, I can automatically route people over to one of the replica servers. As for load balancing, if I have a cluster, I can distribute the load for a group of servers across the cluster. When you see this kind of behavior, you almost take for granted the features in most of the enterprise software that Lotus has been trying to develop.

Tivoli Analyzer for Domino 6 is a good example of what we call the Predictive level of autonomic computing. This tool can look across a group of Domino servers and suggest to you things that you should be doing to make the servers behave better. These may be fairly mundane things, like memory a particular server needs to work better. Or it may be as complex as analyze this set of Domino servers and find the one which has the most free space, then make a recommendation for moving my databases around so that I have better distribution of databases. Then develop a plan that can be carried out automatically through the Notes Administration Process.

With Sametime Enterprise Meeting server, are we seeing some autonomic features with the server's ability to determine where to send or host a meeting on a number of servers?
Yes, that is a good example of self-optimization through load balancing.

Can you give customers a sneak peek of what they can expect in upcoming releases of Lotus products that demonstrate autonomic computing?
We'll see some examples in Lotus Workplace. Even in our first release of Workplace Messaging, we see the ability to auto-provision mail users (that is, to automatically create mail accounts for new users). This is an example of self-configuration. Instead of having to go through what may be a difficult or time-consuming effort to get the right software to the right people, the software does all the provisioning.

Let's talk a little bit about the Lotus Notes Smart Update feature. Is it an example of autonomic computing, and if so, how does it fit into IBM's vision of autonomic computing?
Lotus Notes Smart Update is another good example of self-configuring software. You can push out changes to the client from some central place and that makes it especially efficient when you have tens of thousands of clients that have to be upgraded in a company. That feature makes it far less costly to upgrade. That's definitely a good example. Now where does that fall on the maturity level? It's fairly focused in scope, so we probably put it at the Managed level. There's a lot of automatic operation there, but it only applies to Domino.

I think over time we want to see more attention to solutions (especially heterogeneous solutions) in which you can apply not just one piece of software from one of the brands, but collections of software that work together across all of the brands.

How difficult is it to build an autonomic computing system when you have that level of complexity with different pieces of software trying to work together? For instance, how do you ensure that when something goes wrong with Domino it's not affecting WebSphere because it's running on the same machine?
This is really the big technical challenge. The short answer is we don't really know how to do it today. This is why the IBM Research teams are so busy thinking about some of these issues. To go back to the Log and Trace Preview example, there's an opportunity there if there's standardization of logging to do cross-product and cross-server diagnosis. I think the key really is standards. We can see, for example, the open grid services architecture—the OGSA standard—as being an important way to standardize the way that these autonomic elements begin to communicate with each other.

You mentioned that having that cross-product solution is a huge challenge right now. What are some of the other challenges that IBM is facing?
There are a lot of technical challenges that are pretty daunting in some respects. Think about especially large-scale autonomic systems with tens of thousands or hundreds of thousands of computers or devices that are all somehow working together and self-optimizing to some extent, so perhaps some of the elements are being selfish and are making trade-offs against other elements. We don't know how to build and test systems at that scale yet. So that's one of the things that we definitely need to do some research in. And, of course in the meantime, we're trying to take the baby steps. There's a lot of basic or foundational things we can do before we worry about that.

There's also a social issue, which I think in some ways is just as difficult as the technology issues. We're asking IBM teams and the industry for an unprecedented level of collaboration. We really need for these heterogeneous systems—multiple computers from multiple vendors—lots of software from lots of different people to work together. If it's all going to work, you really need to get people to buy into the right standards. That's not a technical problem; that's a social problem.

You mentioned different levels in autonomic computing, like the Managed level and the Predictive level. Can you describe the levels that make up autonomic computing?
Sure. A lot of this is covered pretty well on IBM's **Autonomic Computing Web site**, but there are five levels. At the

Basic level, individual systems are managed manually and require highly skilled IT staff. At the Managed level, information can be collected from different kinds of servers, which allows for more central management. At the Predictive level, the system is monitoring itself and suggesting improvements to the system administrators, who then approve the suggestions. At the Adaptive level, the system takes action automatically. And at the Autonomic level, all parts of the system are dynamically managed based on business rules or policies.

Some autonomic computing research is happening at the academic level. I know that IBM is involved in programs in certain universities around the country. Can you talk a little bit about what's going on in academia?
There is quite a bit of activity. We talked about the notion of hundreds of thousands of interacting machines. There's a fairly active research genre which is agent-based computing or agent-based modeling. There's also work in the area of complex systems—complex adaptive systems in particular. We hope from some of that work that we may see over time basic design principles emerge. We really need these principles to understand how to build these large-scale systems out of interacting components so that we see stability. There's work I think at the University of California, Berkeley from David Patterson in the area of recovery-oriented computing to begin to help us think about self-healing approaches.

It's a slow process of technology transfer. It's not all academic-oriented. A lot of the IBM research teams are actually very practically focused, and in fact, some of the technologies have moved away from the research teams toward customers. There was a recent announcement about technology that can do load prediction for WebSphere servers. Not only can it monitor response times and detect when load spikes are occurring, but it can also bring a new WebSphere server online to deal with the load. When the load has passed, it can take the WebSphere server off-line and put it back into the spare pool. These are things that are real. We can see them work and can get them into customers' hands.

Autonomic computing impacts the hardware, the operating system, and now the software. It seems to get more and more granular.
Yes, I think so. There will be multiple levels of granularity. And this is one of the common things you see in almost any complex system. There are levels of hierarchy. In the human body, the cells make up the organs and the organs together form the body. In computer systems, we expect to see a fairly low level of control that's auto-tuning available memory, and at the high level, we have policies that may influence how computing resources are shared between divisions of a company.

I'm sure IBM is not the only software company in the market interested in autonomic computing. Do you have any idea what Microsoft is doing to compete in this area?
It's interesting because Microsoft has announced an autonomic computing initiative, which is in some ways good for us because they seem to recognize the value of autonomic computing to the extent that they're trying to take it into account in their own product plans. And Microsoft isn't the only one. I think a lot of the hardware vendors—the larger vendors, like HP—see the same problems in their customers' environments. If you're going to sell a lot of machinery to a company, someone has to administer it. So the drive is to lower their cost.

We think that IBM has an advantage here because what IBM has to offer is so broad—from hardware to operating systems to software. If we pull it off and it all works together, then what IBM has to offer is much broader than say what Microsoft offers in the application space.

How are we working with some of our competitors to advance the technology?
We work in the Web services space to define standards, and again, all successful autonomic computing is really about standards. It's no different than anything else in that respect. We work with competitors to define standards so that we won't be competing at the low level. We'll compete at a much higher level.

What other industries will be affected by autonomic computing? For instance, you mentioned hardware. How are other industries affected by this technology?
I think it's just hardware and software. I'm not sure what else there is from our perspective. There's a lot of opportunity for autonomic behavior at the hardware level. And in fact, a lot of the IBM hardware has some of that now. The machines are self diagnosing. If a board fails and the little lights can come on and tell you, "Oh, these are the ones that need to be replaced," you can get the machines running more quickly.

How will autonomic computing change the software industry?
I think one of the things that changes for the industry is the engineering methods used. Today we have a fairly good understanding of how to build software, and we go through the usual design phases. But as the autonomic vision becomes realized, I expect to see some fairly fundamental changes to the way we design software. How do we design for stability in large-scale systems of millions of interacting devices, not all of which are server computers? I can imagine that the character of testing will change dramatically. You might see simulation playing a greater role. It's simply impractical to buy as many machines as you need to really exercise large-scale autonomic behavior.

From an industry perspective? We hope that one of the changes that occur is that software can actually deliver the kind of value that people are expecting. I think that when you buy a collection of software from anybody, you really expect that it will work together. When it doesn't, it gets in the way of running the business. We hope that one of the major shifts is that it gets people back to the focus of what's important to them, which is getting their business running, and away from what's not important, which is micromanaging all of their server issues.

Is autonomic computing part of IBM's on demand initiative?
Yes. On demand systems need to be resilient. Autonomic computing helps there by reacting to changes like load spikes. IBM is characterizing the on demand operating environment as integrated, open, virtualized, and autonomic.

About how far off are we from fulfilling IBM's vision of autonomic computing?
That's a tough one. Predicting the future is always risky business in this world. When Paul Horn first announced the autonomic computing vision, he talked about it as a grand challenge. It wasn't something that was just difficult and that was just going to take time. It was something that we didn't know how to do fundamentally. We have to learn quite a bit in order to realize the vision. So it's more of a journey, and it's really, I think, difficult to predict when the journey's over.

ABOUT VAUGHN ROKOSZ
Vaughn Rokosz is a senior software engineer in the IBM Software Group (Lotus software). His interests include enterprise deployability, autonomic computing, performance engineering, RAS engineering, complex systems theory, and software project dynamics. He received a BSc in EE and ChE from the University of Michigan. He is a member of IEEE and ACM.