

MATHENY: Welcome to this IBM Rational Podcast. I'm Angelique Matheny. Joining me for this podcast, Delivering Next Generation Converged Applications with Speed and Quality, is Derek Baron, Worldwide Rational Communications Industry Offerings Manager, and Jim Conallen, a software engineer on the IBM Rational Model-Driven Development Strategy Team.

They will explore the challenges of delivering converged applications and the solutions offered by IBM Rational Software. Derek and Jim, welcome to the podcast.

BARON: Thank you. Thanks for having us.

CONALLEN: Thanks a lot.

MATHENY: We've got a lot to cover today, so let's get started. Derek, we'll start with this question. What are converged applications, and why do we need them?

BARON: These are an exciting new breed of software systems that help us to live and work more effectively. I'll give you a couple of examples.

So on a personal level, these applications allow us to place calls that include video streams, to see if our friends are available and then invite them to a party, or to go shopping to keep a respectful and perhaps subtle eye on the location

of our kids.

For business men and women, we'll see connecting with and sharing information become much more seamless and efficient.

Some service providers are already offering these innovative revenue generating applications while others are falling behind. These applications leverage fundamentally video voice and data protocols in combination, and they unify all forms of human and device communications in context.

So to be a little bit more technically precise, a converged communications application manages two or more video voice and data related protocols. And they also maintain session among them, and almost always they're going to have to support a high availability environment.

MATHENY: So why aren't we seeing a lot of these applications in the marketplace?

BARON: Well, great question, and I think the answer starts in the network. So traditional networks handle video, voice and data in a stovepipe, often using completely separate network infrastructures. And this makes building these kinds of applications very difficult and costly.

Building a converged voice and video app, for example, requires a highly-specialized point-to-point integration

process. Doing these kinds of one-offs on traditional networks is very costly and it almost always involves a lot of risk.

So this is why IP-based next generation networks are so important. IP-based networks provide a common protocol for video, voice and data. Additionally, these new network architectures create horizontal planes upon which to build applications.

So we see companies like IBM, Ericsson and many others provide application servers that operate in that services plane. And this is good for service providers, because they get an execution platform that they're already experienced using, and most include SIP support that they need to actually make these applications a reality on their network.

MATHENY: Derek, you just mentioned SIP. What is SIP, and why is it so important?

BARON: Yes, so SIP stands for the Session Initiation Protocol. It's a signaling protocol used to negotiate and modify any kind of media session in an IP network. In fact, Java has standardized on the way to build SIP-based applications. JSR 116 is the SIP servlet specification, and it defines how SIP applications written in Java work on application servers.

In many ways, SIP is to IP-based networks as HTTP is to the Internet. And what I mean by that is that HTTP is what made the Internet mainstream. It enabled Web pages, which dramatically accelerated the adoption of the Internet by the masses.

So SIP is very much like that but for these converged communications applications. It's a simple, easy protocol that makes really cool things possible. SIP...in fact, SIP was explicitly designed to be similar to HTTP. And so, wisely, the Java designers modeled SIP servlets after HTTP servlets, the way that you program these things.

The intention is that SIP servlets and HTTP servlets can actually work together in what's called a converged container. You may not know it, but IBM's WebSphere Application Server is a converged container that supports both SIP and HTTP servlets.

And we have Jim with us, who we haven't heard from yet. But, Jim, you know, you've had a lot of experience with customers trying to manage their architectures when SIP and other services and protocols are involved.

J Yes, sure have, Derek. And you know, the key for every one of them is in understanding what they have in

place and where they want to go with it. You know, architects have always had a good view of understanding their systems at a relatively high level. That's where the scope of an architect's responsibility usually is.

But when it comes to converged systems, and for the communications industry in particular, we need to have a deeper understanding of some of the details of what's going on.

You know, the days of trusting your equipment vendor and the black boxes that live in your network are over. With the move to IMS and generic IP networks, everything these boxes do is important.

So it is critical that you know what is going on, and what they're doing. Do they require DHCP? How often do they ping other boxes? Do they require a DNS? And what frequency are they hitting it? You know, what happens during bootstrapping?

These are all issues that affect the scalability of the network and the system as a whole. Where we, IBM Rational, come in, is in providing some tooling that helps architects to visualize and understand the totality of the behavior of their system.

For example, simple UML sequence diagrams are an excellent way to visualize the dialogue of activity between the various components over the network. Of course, documenting the details of all the messages being generated by a system itself has scalability issues that need to be addressed.

Fortunately, the current versions of our UML modeling tools enable architects to craft these sequence diagrams in a way that you can reference other even shared interactions inside your diagram.

What this effectively means, that you can document standard things like DHCP, authentication, in a reusable asset and then just refer to it in your sequence diagram. The full details of the DHCP authentication or the SIP invite, you know, you can always drill into them. However, in the context of your current diagram, it appears to be just a simple reference allowing you to focus on the overall flow of your stuff.

Our modeling tools have a similar mechanism for the other UML behavioral diagrams like activities and state machines, both of which are very useful diagrams for architects to document and better understand the systems and the behavior of the systems that are going on in their system.

Now, if the SIP services you are developing are already

running on the network, then we have some more tooling that can help. The latest release of the SIP modeling toolkit has a new transformation that lets you import PCAP files with SIP traffic in them. And it will produce full call flow diagrams, sequence diagrams with SIP headers and body content in them.

And a PCAP file is a standard file format for the capturing of network packets. And there's plenty of free and available software out there that can help you produce these files.

This new functionality essentially lets you record SIP traffic and visualize it in your model. And once it's in the model, you can modify it, share it, and even make it a reusable asset that can be referenced by other diagrams.

MATHENY: Jim, tell us more about that SIP modeling toolkit.

J You need to be careful with that question. See, I've been involved as a lead on this project for close to two years now, so I can talk to you about it for hours. But don't worry, I'll provide the abbreviated overview.

The SIP Modeling Toolkit is a freely downloadable set of extensions to the Rational Modeling Platform designed to be

of use to developers and architects building SIP services on WAS or any JSR 116 compliant platform.

Its feature set is divided into three main areas: call flow modeling, servlet modeling including HTTP and that WAS converged servlet and [SIP lister container lip service], and model to code and test case transformations.

So most of the interest we've seen in the toolkit so far has been around call flow modeling support. We've extended the standard UML sequence diagrams to allow it to capture the details of SIP and HTTP messages -- that is, you know, the headers, their values, the body content of the messages.

Making these sequence diagrams, call flow diagrams that contain as much of the details of the SIP and HTTP messages as you'd like.

We've done our best to make the editing of these diagrams as easy as possible, including the import of pcap files, as I mentioned earlier, to show the actual real life call flows.

Now, in the toolkit my favorite new feature is the ability to merge multiple call flow diagrams into a single state machine which represents the emergent behavior of the combined call flows.

Now, this is not the most important feature, by far, but it

really is an interesting use of modeling to help gain an insight into an evolving design. By combining call flows, individual call flows with separate scenarios, you can begin to see the level of complexity that that resultant service might be taking.

And it's far better to figure out and understand what's going to be in there in the abstract modeling world than to only find that out after you've already built much of it and invested a significant number of resources into it.

Call flow diagrams are often the primary input to the design of a servlet, a SIP servlet or HTTP servlet. Call flows can be extended further to document the internal behaviors of the elements that implement the service. And these would include the servlets and all their supporting classes -- thereby you're establishing a connection between the design of the service and the call flows that are driving its specification.

Our servlet modeling extensions are just your basic stereotypes on top of UML classes and interface elements. And by tagging the servlets appropriately, you can edit their SIP and HTTP specific information so that during code generation the projects deployment descriptors are also updated.

Now, another new feature of the toolkit is the ability to import most of the significant content of SIPp files. SIPp is an open source SIP traffic generator that's very popular and easy to use and configure. It's a great source of just generating SIP traffic and SIP messages.

So, SIPp is not meant to document the entirety of the call flow; we do our best with some heuristics to reconstruct what the architecturally significant aspects of the flow might be.

But the flow in the model, you can add or remove some detail, and then use another new feature of the toolkit to generate a Rational Performance Tester for SIP test case. So, once RPT for...once [within] RPT for SIP, the test case is updated to use any available data pools, and you can add verification points.

And now you've got a first class test case that can be scheduled and run to do both performance and functional testing of your SIP and HTTP service.

So what we now have is a set of modeling abstractions that are used to help us understand the service under development, and that are also used to drive the implementation and those same artifacts are used to drive their tests.

And having all these important analysis and design artifacts integrated into the same platform really makes for an efficient development environment.

MATHENY: Thank you, Jim, for that abbreviated version. You could go on and on.

J [LAUGHTER] Yes, sorry about that.

MATHENY: Derek, I'll close with this question. Can you summarize the IBM Rational support for converged communications applications?

BARON: Yes, absolutely, you bet. Jim did a fantastic job talking about the SIP Modeling Toolkit, and I'll touch on the testing tool that he mentioned as well.

To summarize, for architects and designers, we offer a SIP Modeling Toolkit. And as Jim said, it's a free downloadable plug-in for owners of Rational Software Architect or Rational Software Modeler.

For developers, Rational Application Developer -- which is by the way also part of Rational Software Architect -- it comes prepackaged with a number of SIP wizards and editors to help developers write SIP-based applications.

Users are also able to download for free the Telecom Web Services Simulator, which is also an addition to...that you can add into Rational Application Developer or Rational Software Architect.

And what that does is that provides developers with a simulation environment for developing a telecom Web services that meet the Parlay X Web services standards.

For testers, we released a SIP protocol extension to our Rational Performance Testing tool, and that's the RPT for SIP tool that Jim was talking about earlier. So RPT -- or, Rational Performance Tester -- supports other protocols in addition to SIP, things like Web Services protocols, SAP, HTTP and others.

But before I...before we leave here, I just wanted to mention, you know, there's this age-old paradox in software development, and it's called speed versus quality -- essentially meaning that if you build something faster, the quality suffers. On the reverse side, if you focus on quality, it can often take too long to deliver an application.

So Rational Software is designed to help you overcome this paradox and accomplish both objectives. And we've had a lot of success helping companies to do that. The key is

really...to achieving this is to combine individual productivity improvements with improved workflow across the workflows. And we call this unifying the team, and we do that with integrated tools across the entire life cycle.

So if you're going to be building these SIP-based converged communications applications, we provide very specific productivity enhancing features in three areas: in the modeling with the SIP Modeling Toolkit, the coding environment that's available in Rational Application Developer by default, and then the SIP extensions to our testing tools.

And they're fully integrated, they work well together, and it's a very nice environment. So I think with that, I think that about sums it up, Angelique.

MATHENY: Thank you so much, Jim and Derek. That was a really great overview. And I appreciate you taking the time out to discuss delivering next generation converged applications with speed and quality. We really appreciate it.

J Thank you very much. I had a great time.

BARON: Enjoyed it. Thank you.

MATHENY: That was Rational's Derek Baron, Worldwide

Rational Communications Industry Offerings Manager, and Jim Conallen, a software engineer on the IBM Rational Model-Driven Development Strategy Team.

For more information on this topic, we have a Webcast that was delivered on May first and is available for replay. It features Derek Baron, entitled with the same title: Delivering Next Generation Converged Applications with Speed and Quality. Please visit the developerWorks Webcast page for the listing.

If you're interested in more podcasts like this one, check out the Rational Talks To You Podcast page at www.ibm.com/rational/podcasts. This has been an IBM Rational Podcast. I'm Angelique Matheny. Thanks for listening. Keep tuning in as Rational Talks to You.

[END OF SEGMENT]