John Paganetti: Reliability Guy

Interview by
Barbara
Burch

Level: All
Works with: Domino 5.0
Updated: 07/01/99

Inside this article:

Related links:
Domino R5: The Domino Internet Cluster Manager

Russ Holden: Domino R5 Database Improvements

Measuring your Domino server's reliability

Lotus Performance Zone

Get the PDF:
Kb

*[Editor's Note: To learn more about Domino R5 server reliability, check out the discussion with John Paganetti in the **Developer Spotlight**.]*

Fresh from receiving the 1998 Lotus Commitment award for outstanding service, John Paganetti personifies reliability. He spent many weekends and all-nighter's, working toward the goal of 99.6 percent uptime for Domino R5 servers. Here's the story behind the R5 scalability and reliability numbers.

Server reliability and availability were a big part of Domino R5. What was your role in achieving this goal?
Well, let me start off by saying that no one person is, or possibly could be, responsible for the quality and reliability of Notes/Domino. It's an on-going, massive team effort that benefits from years and years of experience of building quality and reliability into the product. We've fine-tuned the whole Notes/Domino coding environment over the past 15 years to help us detect and debug problems in a very timely fashion. When developers add new code to our source pack, they immediately get to take advantage of many error detection features that are built into our core product for free. Everyone is also highly encouraged to add further debugging and/or tracing features into their code to help troubleshoot problems as they may arise.

A great part of the success of Notes/Domino is the fact that the product itself makes it tremendously easy for developers to share their wealth of knowledge. It's the old "which came first, the chicken or the egg" discussion. That is to say, without Notes, Notes development would not be possible today.

But back to your question. Just prior to Lotusphere '98, I had a meeting with Steve Beckhardt, the president of Iris. He said that he was very committed to server reliability for R5, and based on my past experiences with server reliability for R4, he asked if I would be a so-called "server czar" for R5. The goal was to make sure that R5 was the most reliable release of the Domino server ever. We set our R5 reliability goal at 99.6 percent uptime, measured over a two-week period. The ultimate goal obviously is 100 percent, but we know from past experience that 99.6 percent is a very acceptable uptime. Personally, I set the goal to achieve 99.8 percent, because in my mind, this would be more than acceptable and pretty amazing!

You mentioned your past experiences with reliability for R4. What were they?
When I started at Iris more than five years ago, I initially worked on the porting of the Notes server to NetWare. This involved spending time in the OS subsystem of Notes, figuring out what had to be done to get a certain product area functioning. For example, when something wasn't working right in the Replicator, I had to dive into that area. Or, if the Router wasn't delivering mail properly, I'd have to look into that... This allowed me to become familiar with many different areas of the server. I also learned after a while how to distinguish whether a problem was platform-specific or cross-platform, which obviously affected what code needed to be changed to resolve the problem.

I ended up spending quite a bit of time in the server lab while working on the NetWare port, so I also started keeping an eye on the other servers as well. When they had problems, I'd give it my best shot to figure out what was wrong. But usually, I'd end up chasing down the appropriate developers and watching over their shoulder as they debugged, so I could pick up their troubleshooting tips and tricks. As we did further ports, I learned about more operating systems and their debuggers.

Over the years, you also become intimately familiar with the whole development process. Unfortunately for me, but fortunately for our reliability, I spent quite a bit of time here at Iris. You learn from experience that if you get in early enough in the morning and help "clean" the morning build, you'll know what changes went into the build that may end up being problematic.

What does that mean for a build to be "clean"?
A build is "clean" when we've made sure that the entire Notes/Domino product, both client and server, compiles and links successfully on all platforms, and that no new code has "broken the build." This is another great strength of Notes/Domino as far as reliability is concerned. Running the code through many compilers nightly helps us to quickly identify bugs in the code before it ever gets into the hands of a single user. Any code that required a change to get the morning build "clean" is immediately suspect and stays fresh in your mind. As soon as we've quickly smoke-tested the daily builds, we put them on several Iris production servers. Then, if any problems arise, it's usually pretty easy to figure out what change from the previous day may have caused the problem. It's also easy to identify through our source control tools who introduced the change.

As I'm sure many people are aware, anything less that 24x7 server availability is unacceptable. So, when we reached the end game, we spent quite a few late nights and weekends monitoring the servers as well. This constant vigil allowed us to really get a feel for the areas of the product that were doing well, and the other areas that were at risk. We were then able to

ask the key developers to help solidify these areas before we shipped the product.

How were you involved with the internal Beta deployments of R5 -- across Iris, Lotus, and IBM?
To start off with, just the growth of Iris itself was great for R5. Back in R4, Iris was only 70 employees and by the time R5 shipped, we were almost 400. We not only had more production servers in the Iris domain, but we were able to put a much higher, real-world load on them. This enabled us to do a lot more live debugging onsite, where we could track down developers almost immediately. Notes/Domino Quality Engineering (QE) became a part of Iris early on in R5, and with this addition came a new Notes domain. This again increased the number of production servers that we had at our fingertips for R5. I would personally like to thank Dave Kelley and his team of administrators for all their help loading daily builds, chasing down developers, and writing Software Problem Reports (SPRs). If I had a nickel for every developer we paged to the main server lab over the course of R5 development, I probably wouldn't be here for this interview!

How about the Beta deployments within Lotus?
We also had a massive Beta deployment within Lotus, and worked closely with Lotus Information Services (IS) in Cambridge. A selected number of servers were ear-marked for the deployment, ranging from mail servers for Lotus IS personnel, Sales and Support, and even the servers for the President and the top executives at Lotus. Believe me, you would immediately hear about it when any of these servers had a problem! For R4, I was the Iris liaison for working with Lotus IS. It was the first time that we actually had a major deployment plan, and we shook out many of the problems with the process over the course of the deployment. I still recall taking many a late night car ride to Cambridge to investigate and debug problems. Remote debugging was possible at the time, but there's no substitution for being there live, because even the slightest observation can give a clue to resolving a problem.

Greg Pflaum took over this Lotus liaison role for R5. I know that it's been over-used quite a bit this last year, but Greg gave a truly "super-human" effort. He was tenacious about resolving problems that came up during the Lotus deployments. During my time with the R4 Beta deployment at Lotus, we rolled out R4 to approximately 10 production servers about three months prior to shipping the product. Under Greg's command, we rolled out R5 to about 15 production servers in Cambridge, and some worldwide servers as well, starting approximately six months before we shipped.

Then, how about IBM deployments?
Well, IBM was the new kid on the block for R5. IBM set up a Center of Competency Center (COC) in a building adjacent to Iris. They built a 24-hour smoketest that attempted to simulate as best as possible most of their IBM production servers. Each and every daily build had to pass this 24-hour smoketest before deployment on any live IBM production server. The smoketest used live replicas of the IBM Domino Directories, and I believe that the U.S. version of their directory contained approximately 350,000 users. So, this was a great test environment for many of our scalability issues. Since this was a brand new Beta deployment initiative, it took us a while to establish all the necessary chains of communication. We were able to deploy R5 on one IBM production server, starting approximately two months before we shipped. It may not sound like much, but it was a huge success for us. We were able to work in a whole different environment, and it definitely helped us to nail down some scalability issues before we shipped.

How do you think this experience will affect future deployment efforts?
We've really laid the groundwork for future deployments at IBM. Like in R4,

what we did with Lotus IS was successful, but what we really did was lay the groundwork for what we ended up doing for R5. We got the right contacts and procedures in place, and so on. This led to a much larger and smoother rollout for R5 within Lotus. So, we expect to have a more pervasive deployment throughout IBM for future releases, starting with R5.0.1.

But isn't IBM deploying the R5 Gold release now? How's that going?
Yes, IBM's busy deploying R5 across their mail hubs, mail servers, and so on. For mail hubs, they've deployed to about 10 hubs in the U.S., and five more across Europe and Asia. Then, there are about 13 mail servers in the U.S. running R5, and four more in Europe and Asia. They have also deployed other R5 servers, such as a Domino Directory server and a semi-production HTTP server. The users on the mail servers range between 300-500 users, with peak user loads around 200. One S/390 server has 1500 registered users, with peak user loads around 700.

And, how are the reliability numbers looking?
Well, I can speak better for the 20 servers that I monitor in the Iris domain. For those servers, our uptime since loading the actual R5 Gold build onto the machines has exceeded even my wildest expectations. It's just been tremendous. For three weeks now, we've been running the Gold build and we've only run into a single problem on one machine, so it's been approaching 100 percent uptime.

To put the numbers in perspective, and forgive me if my math betrays me, but a 99.8 percent uptime over a two-week period figures out to be about one crash per server where the server was down for about 40 minutes. Or, you could have two server crashes per server where the average downtime was about 20 minutes for each crash. We had just one server crash on all 20 servers over a three-week period. Now granted, the user load on many of the servers was minimal, since there are only about 400 Iris employees, and much of the load is spread across the servers so we can get some coverage for all the different OS platforms. However, several of the heavily used servers consistently ran with between 150 and 200 active users and experienced no problems. If I compare these results with what I experienced and recorded for R4, R5 is by far a much more stable and reliable build.

How did we get to this point of approaching 100 percent uptime?
Well, it took a lot of work by a lot of people. One of the key things that I formed this year was called the Domino Server Reliability team, which would meet once a week at lunchtime. The team included key developers in the key areas, and also Tim Halvorsen, the Chief Technology Officer (CTO). We met for about an hour each week, and went through all the crashes that we had seen. We would talk about what crashes we had seen that had been fixed, and what we did to fix them; and what crashes we had seen that we didn't fix, what people had been assigned to them, and what SPRs had been written up. We brainstormed about how developers could resolve crashes without waiting for them to happen again. When we saw patterns developing across multiple crashes, it helped us hone in on certain areas. We covered the Iris, Lotus, and IBM deployments, and these turned out to be very productive meetings.

In addition, we did a lot of smart things with clustering within Iris, Lotus IS, and IBM to allow us the extra time that developers sometimes need to troubleshoot problems on the server. We configured the R5 servers in clusters with other R4.6 servers, so if the R5 server went down, the users would failover to the other server. So, the users didn't lose service, and we didn't lose the debugging session. This allowed us to be much more aggressive in deploying new builds daily into live production domains, and to quickly identify any new problems that occurred.

Your name's associated a lot with clusters. Why?

I didn't do the original coding for clustering, but as members of the clustering team moved on, I took over the role of supporting clustering in R4. As things turned out, I became a key liaison between Iris and the developers at IBM who signed on to do most of the clustering work for R5. We brought IBM in to take advantage of their expertise and personnel. So, I helped educate them about our existing clustering code and our whole development methodology. I worked closely with Mike Kistler and his team from IBM Austin, and we have an ongoing relationship with them. His team did a great job with the new development and QE of clustering for R5. And as we move forward, we will definitely be further leveraging IBM research and a lot of the information that they have gathered about clustering. That's one of the strong points about our relationship with IBM -- being able to take advantage of their expertise and knowledge where appropriate.

What were the clustering improvements for R5?
The main new cluster feature for R5 is the Internet Cluster Manager (ICM), which essentially brings the Domino clustering capabilities of the Notes client to Web browsers. You're able to load balance, failover, and find the best available server by having your Web browser connect to the ICM first. The ICM essentially redirects the client to the best available server. Although the ICM was the big piece, we also made some performance improvements to the existing code in the clustering logic. These improvements were based on observations from our internal deployments, as well as a few customer sites. Of course, we also fixed a few bugs along the way.

What does the Internet Cluster Manager (ICM) bring Domino in the marketplace?
Before R5, there were, and still are, solutions available like TCP/IP sprayers to attempt to share the load within a cluster, but they don't have any real smarts. They would basically just spray users to different servers, but they didn't possess the knowledge that Domino clusters have. Plus, in the setup of Domino clusters, we recommend that you limit the number of database replicas to only those that you need to make highly available. For some databases, this may mean that you have a replica on every server, but for others, this means only two replicas, no matter how many servers are in the cluster. So, if you use a TCP/IP sprayer, there's always the chance that the sprayer could send users to a server that doesn't have the desired database.

Now, if we send users to the ICM, the ICM has the same smarts as the Domino cluster, such as what servers have a particular database. Also, you can have some servers in your cluster running HTTP and some running HTTPS. The ICM is intelligent enough to know the servers that run securely and the servers that don't, and to send the user to the appropriate server running the appropriate protocol, and also containing the desired database, all transparently. In addition, the ICM can do things like true load balancing because it also contains the smarts as to what type of load occurs on each server. It can send users to a server with a lighter load, when appropriate. A sprayer doesn't know this, so it just sends the user off to the next server in line, despite the server's current load.

Plus, you're able to just put the ICM in place. It's something out-of-the-box from Domino that end users can use without requiring any other hardware or software. In addition, we have means of making the ICM highly available. [Editor's note: For more information on the ICM and how to set it up, see "**Domino R5: The Domino Internet Cluster Manager**." For more setup information on clusters, see the **Domino 5 Administration Help**.]

What's in the future for clustering?
Well, there are a lot of things on the table. For example, right now, we only failover on database open. Our future direction is going to truly make the user experience transparent. Things like when you're editing in your mail

file and your server goes down, you currently have to wait for the server to come back up, so you can save off your document. If you want to open your mail file and your server is down, we fail you over properly because it's on the database open. If you're in the middle of an Office Notes database, and you're scrolling down when the server happens to go down, right now, we don't automatically reconnect you to another server that happens to be up, and put you in the same navigational position as you were. And that's basically because there's a lot of work that needs to be done in that area to allow us to do that. It'll be a big chunk of work, but people definitely want it. It's probably the number one request -- to be able to failover more appropriately in the middle of a session, as opposed to just when a session begins.

Also, people may not be aware that we have a great MSCS (Microsoft Cluster Service) story today. Even with R4.6x, we supported an Active-Active, two-node MSCS cluster by taking advantage of Domino partitioning. To me, a really cool solution is combining MSCS with Domino clustering using Domino partitioning. This gives you the best of both worlds. I thought it was great that we could support an Active-Active model for our Domino service on MSCS years before Microsoft Exchange could, but I'm sure they'll have that capability someday soon as well. We're looking to further improve our integration with MSCS in areas such as install and administration. We'll also be exploring other OS clustering solutions as the market demands it.

Let's get back to reliability. Was the goal of reliability a result of pressure from the competition, user requests, or what?
It's really a combination of both. For end users, their whole impression of the company and the product is how much uptime they have, and whether they can access their mail files. In addition, there's always competitive pressure. If your client goes down, you affect one user. If your server goes down, now with our scalability story, you could be affecting upwards of thousands of users. So, it's a huge, competitive thing, and I'm very pleased with our current situation with R5. I know that customers deploying it in the field are going to be very satisfied. There will always be some problems that slip through the cracks. Realistically, you can't test everything in-house, but this was by far the widest Beta deployment that we've ever had. Believe me, we fixed a lot of bugs that customers will happily not have to deal with.

When's the final "clicking" point -- when do you know how reliable of a release that you really have?
What happens with a major release is that new features come in left and right, and until we actually get to the point of code freeze, the servers are going to be very unstable. A ton of fixes (both client and server) go in up until the final push, and the one thing that you want to reduce when you're going for reliability is the rate of change. As we moved closer and closer to shipping, we more heavily triaged what was allowed into the build, and restricted the number of changes being put into the server. Any server crashes, obviously, were fixed in the release, so even as the number of fixes allowed into the build slowed down, the number of these fixes that improved server reliability actually went up. So, as the rate of change changed, the reliability numbers went up.

So, how does our reliability compare to that of Exchange?
To be honest, I don't focus on the reliability numbers of our competitors. My job is more of a heads-down effort in attempting to achieve a magical goal of 100 percent server uptime. I know one thing for sure, our server reliability numbers for R5 are not going to give them any ammunition to use against us. One of our strengths in the recent past has always been the greater reliability of our Domino servers as compared to Exchange servers. Trust me, it takes many years and lots of discipline to build a highly reliable messaging/groupware server, especially with all the new features being

added into each major release. At this point, we've completed five major releases of Notes/Domino. The bottom line is that you have to go through your growing pains. Reliability wise, I'd put our server up against theirs any day of the week..

Now let's talk scalability. Domino's scalability, and its reliability when it scales, continues to be a competitive advantage for us. Believe it or not, the scalability improvements that we've made in R5 have made the server even more reliable. You would figure -- more users, more of a chance of a problem. But, I can think of two specific features that we did in R5 for scalability reasons that also improved our reliability: transactional logging of our databases, and the use of thread pools to allow us to scale to more users per server.

Can you talk more about how the database improvements in R5 helped with reliability?
Well, one of the major problems, as far as reliability of the server goes, has always been data corruption. A single corrupted database can wreak havoc on a server's reliability numbers. Proper error handling is crucial to any server's reliability. I am proud to say that our core product's error handling is excellent. However, it's still true that we write and test most new code to deal with good data. When the code encounters bad data, you exercise your lesser tested error paths, and many times your server crashes because of a bad error path. Unfortunately, you usually don't find the bad path until it's too late and you've brought down the server. So, a lot of the times when we do get crashes, we find that they're on error paths and error conditions, or that bad data was returned from a corrupted database. Also, it doesn't matter how reliable the server is if the data you're trying to access is corrupt and inaccessible.

Russ Holden's database team had a tremendous impact on the reliability of the server. Their work helped to dramatically cut down on database corruptions in comparison with R4. The new ODS (On-Disk-Structure) for R5 not only performs better, but key components have also been made redundant to reduce data loss and improve data integrity. We do more integrity checking of data before it ever writes out to disk, assuring that it doesn't corrupt the data already stored. And when data is read back, we verify that the data read back is good as well. Transactional logging, and the speed at which a server can be restarted, is vastly improved and deemed by many worth an upgrade to R5 by itself.

How does this compare with R4?
Anytime the server crashed, anytime any database that was open -- say, if you had a 1,000 users with mail files open -- the server would have to run Fixup. And, even though there may not have been any problems with the database, Fixup would have to go through and verify all of the data in the database. These consistency checks could take quite some time to go through, depending on the total size of the files on the server. For example, several Lotus servers have about 100GB of data. It could suck up the CPU to verify all of this data, and people would have to wait for the server to become available. But now transactional logging uses the logs and rolls forward any log operations that didn't write out. So, it doesn't have to go back through the entire database. It will happen very quickly. [Editor's note: For more information on the Domino R5 database improvements, see our **interview with Russ Holden**.]

How does your Mean Time Between Failures (MTBF) tool help administrators with reliability?
The MTBF tool originated back in R4 when we weren't onsite with Lotus IS, and we wanted to get true numbers of how we were doing. People would have gut feelings prior to R4 about how the servers were doing, but it was all based on word-of-mouth. So, you wouldn't know for sure if they were being realistic about the quality of the server. MTBF basically gave us the

ability to monitor any crashes that could occur on any of your servers, so we could say "OK, it sounds normal," or "we're getting way too many crashes." It's a tool that took all the advantages of Domino and replication, and really made it a lot easier for me to monitor what was going on and how each and every part of the build was doing -- whether we were getting more reliable or whether we were going backward on some builds.

We used MTBF extensively for R5 within Iris and Lotus. We will also be using it within IBM as well for R5.0.1. It's not part of the product now, but it may be incorporated sometime in the future. [Editor's note: For more information on MTBF, see "**Measuring your Domino server's reliability**." You can also **download MTBF** from the Iris Sandbox.]

How does MTBF compare with the monitoring features within the Domino Administrator?
The new Domino Administrator will ping your servers and monitor your server tasks. It's more of an online check, making sure that things are live. It will also test your connectivity better than MTBF. MTBF is basically a log scrubber, which looks for server started and server shutdown messages. It does some basic calculations to determine if a server crashed. If it finds two server started messages before a shutdown message, it makes the interpretation that the server must have crashed. In contrast, the Server Monitor within Domino Administrator allows administrators, as long as the connectivity is good, to basically ping the servers intermittently and check even down to the task level that things are running. For example, MTBF won't detect that your router has gone down because the server's still responding. The Domino Administrator can tell you that, and that's a very useful tool. But, it doesn't keep statistics yet. This may come in the future, and it may end up doing a lot of the work that MTBF does.

How does performance testing, and work by the NotesBench group, help with reliability?
Our internal Enterprise Testing team, led by Susan Shaye, did a great job of emulating very large workloads on all the server platforms and eliminated a lot of scalability problems by really pounding on the servers. I know many developers spent a lot of time in Susan's lab troubleshooting problems. It's always good to find scalability problems before our enterprise customers do!

Also, the performance testing and scalability testing by the **NotesBench** group helped us shake out many bugs over the development cycle. They don't have as wild a production environment as we do, where you've got any number of end users, writing their own agents or adding different pieces of code, or creating their own databases. However, their controlled environment is a great smoketest for any new changes that come in. They make sure that no new changes affect scalability and/or performance by establishing a baseline of previous results and flagging any regression. The Beta deployments and the real end-user experience are where we shake out most of our bugs. But, we don't have 10,000 users, so they run a 10,000-user test against the server to prove that we can scale. The scripts that they run aren't softballs. They're very user-intensive mail scripts; they try to emulate what an active end user would be doing. I think that's one of the other strengths of Notes. These tests aren't fluff. You could just open up 10,000 sessions and not do anything, and say "Oh, we scale." But, we actually bang on it, and run scripts against it. We make sure that the server holds up and that every single client is as responsive as the next when we do our tests.

Do you have any reliability or scalability tips?
Your server reliability is only as good as your weakest piece of code that's running. I've talked at Lotusphere about troubleshooting and monitoring your servers. When you're having problems, try removing certain tasks just so you can isolate the problem. But, the less code that you have running on

the server, the more reliable it will probably be. So, we try to look at every single add-in task that people will load and use in deployment, because all it takes is one poor add-in task that could mess us up. Business Partners have to be aware of that as well when they write API programs that run on the server. They have to make sure they're solid as well.

What's next for server reliability and scalability?
We're trying to take a little break from the R5 push, but server reliability is one of those things that you can't let up on, because all it would take would be one bad change that would adjust your numbers. We put R5 gold up on our servers to see how well we really did, and the numbers have been tremendous. It's by far the best major release that we've ever gotten out the door as far as server reliability. I'm very pleased with the reliability numbers that we came up with for R5 -- unfortunately for me, but fortunately for our customers, we've raised the bar with R5 and we're committed to improving on this for future releases.

BIOGRAPHY
John Paganetti has been with Iris for six years as a software developer for the server team.