

MATHENY: Welcome to this IBM Rational podcast. I'm Angelique Matheny. Joining me today is Scott Nordstrom, Senior Project Manager for Rational Software Analyzer, and the podcast titled One Scan, One Analysis, One Decision and Only One Company Has the Static Analysis Framework.

Today we'll learn more about Rational Software Analyzer's extensible foundation for increasing productivity and simplifying static analysis and why we can't live without it. Scott, welcome to the podcast.

NORDSTROM: Thank you.

MATHENY: Let's start with this question, Scott. Can you give me an overview of exactly what is meant by the Rational Software Analyzer Extensible Framework?

NORDSTROM: Yes, I sure can. The overall purpose of having an extensible framework is to have the ability to adapt, adapt to how all developers actually build their source code.

The Software Analyzer Framework allows the developer to expand on their overall quality of their applications into such areas as static code quality analysis, security vulnerabilities, IP management and architectural analysis and visualization, as well as expanding that out into the

application asset inventory.

The sky's pretty much the limit here when you think about actually having an extensible framework and exactly how you can extend that framework into other areas, other than just source code analysis, right?

So what you have the ability to actually go do is extend this out to both things like XML and HTML, and to doing build and installation validation as well as configuration validation just to name a couple of things that the framework will allow you to do.

What we can do through this piece and through this extensible API is to provide the capability to extend the user's analysis through integrations in these areas, such as customizing specific rules or importing or exporting rules and allowing for specific groups to utilize these rule sets in the specific areas around the application today. This can also extend into any automated build system through the command line interface, also allowing for automation and centralization of this whole process.

Every developer pretty much develops differently, right? So what we look at within Rational Software Analyzer is to have an extensible framework that will adapt to how each developer actually develops and what their best practices

are without disrupting the whole development team.

MATHENY: Scott, in the title we say One Scan, One Analysis and One Decision. What exactly do you mean by that?

NORDSTROM: Yes, so on the same track of the extensible framework and having the ability to tie other quality source code tools as well as any type of application analysis solution into the extensible workflow framework.

Most development organizations pretty much utilize multiple source code quality tools in their overall development process today. When you tie all these tools together into one framework, the user now has the ability to do this one scan of the source code, to get one result back from that scan, to analyze the potential application risk associated with that scan and with any other risks that might be within that application and make one decision based on that analysis.

So typically with the scenario of the multiple tools sitting in development organizations, teams are spending a whole lot of time learning and using these tool sets in their day-to-day process when they could be spending more time actually doing development.

MATHENY: So, Scott, how does that benefit customers? How does it help to simplify static analysis and improve productivity? I think that's a pretty important question.

NORDSTROM: Yes, that's a very important question. This allows for static analysis to become more simplified through having the team only learn one tool and utilize the analysis and reporting capabilities of that one tool.

Now what you can do is you can tie all these tools into an automated framework engine and scan, analyze and react to the results in literally a single session, allowing the team to become much more proactive instead of becoming reactive to where quality teams are actually sending the issues back down to the development organization. Thus, you improve the overall productivity and health of that application.

MATHENY: Yes, I can see how that would definitely be beneficial. Are there any customizations that can be done to meet, let's say, specific needs of a customer?

NORDSTROM: Yes, there are. The Rational Software Analyzer extension framework provides multiple Application Programming Interface -- an API -- as well as a whole set of extension points that allows you to customize specific rule sets to meet the needs of what that development organization needs.

What you can do here is you can create your own language parsers for custom analysis. You can create specific groups of rules and customize these rules and views as well as do customize reports through a whole template system that the product actually has.

And lastly, you can create your own data collection points -- meaning that you can collect the data that you want to run during the analysis. This extensibility provides a complete flexibility to meet any of the specific needs that that development organization actually has.

MATHENY: Scott, I'm going to end with this question. What is the overall value of static analysis?

NORDSTROM: Static analysis means the study of things that are not changed. However, in software terms, this definition can be refined as the study of source or binary code that is not currently running -- meaning that the code does not have to actually be compiled to actually run a scan. Thus, static analysis provides the capability for identifying code level issues during the actual time the developer's doing their actual coding, saving development time and overall costs.

Every day we see more and more on how software powers the

modern world that we live in in such things as consumer electronics, appliances, cell phones and automobiles. Also in other areas where networks that we connect to and embedded systems that we tie to, also we see software automating or doing our routine tasks that powers the technologies of our daily lives today.

Software defects cost the U.S. economy some \$60 billion a year, amounting to about six percent of the U.S. Gross Domestic Product according to the National Institute of Standards and Technology. Studies released in 2002 showed that \$22 billion of these costs could be recovered through better software testing.

One aspect of time savings achieved with static analysis tools should be fairly obvious: it takes you less to get better code quality by doing this, again, at the development time. Many studies include some concluded actually within IBM claim that even simple automated code reviews will find anywhere from five to 15 percent of all defects in the code.

The same study also claims that defect reporting by one of our customers costs them anywhere from 12 to \$18,000 per defect if it actually gets out to the field.

If you consider a typically large piece of software that has thousands of defects over its overall lifespan, you quickly realize that using automated code review static analysis

pieces can save anywhere from \$600,000 to as much as \$2.7 million. Regardless of which percentage of cost you actually believe in, the potential savings with static analysis tools is staggering.

MATHENY: Scott, thank you so much. This is really great, learned a lot today. Thanks for taking time out to talk about One Scan, One Analysis, One Decision and Only One Company Has the Static Analysis Framework. We really appreciate it.

NORDSTROM: Thank you.

MATHENY: That was Scott Nordstrom, senior product manager for Rational Software Analyzer. 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]