



[HTTP://www.dotnetrocks.com](http://www.dotnetrocks.com)



Carl Franklin

Carl Franklin and Richard Campbell interview experts to bring you insights into .NET technology and the state of software development. More than just a dry interview show, we have fun! Original Music! Prizes! Check out what you've been missing!



Richard Campbell

Text Transcript of Show #467
(Transcription services provided by [PWOP Productions](#))



Kiciman and Friedman Bring AjaxView to Studio!

July 27, 2009

Our Sponsors





Kiciman and Friedman Bring AjaxView to Studio!

July 27, 2009

Geoff Maciolek: The opinions and viewpoints expressed in .NET Rocks! are not necessarily those of its sponsors, or of Microsoft Corporation, its partners, or employees. .NET Rocks! is a production of Franklins.NET, which is solely responsible for its content. Franklins.NET - Training Developers to Work Smarter.

[Music]

Lawrence Ryan: Hey, Rock heads! Stop paring your pineapple and listen up! It's time for another stellar episode of .NET Rocks! the Internet audio talk show for .NET developers, with Carl Franklin and Richard Campbell. This is Lawrence Ryan announcing show #467, with guests Emre Kiciman and Mark Friedman, recorded live, Monday, July 20, 2009. .NET Rocks! is brought to you by Franklins.NET - Training Developers to Work Smarter and now offering DotNetNuke video training with Chris Hammond from Engage Software on DVD, dnrTV style, order your copy now at www.franklins.net. Support is also provided by Telerik, combining the best in Windows Forms and ASP.NET controls with first class customer service, online at www.telerik.com, and by CoDe Magazine, the leading independent magazine for .NET developers, online at www.code-magazine.com. And now, the man who's dying a slow death on a pub tour of Ireland, but oh what a way to go, Carl Franklin.

Carl Franklin: Thank you very much and welcome back to .NET Rocks! Carl Franklin and Richard Campbell here with you for your .NET listening pleasure. What's up, Richard?

Richard Campbell: You know, I'm home for the summer. It's been beautiful here barbequing, what more can you ask for?

Carl Franklin: Dude, family reunion last weekend?

Richard Campbell: Oh no.

Carl Franklin: All I'm going to say is my father's side of the family is I like to call them educated rednecks.

Richard Campbell: Interesting.

Carl Franklin: Yeah. Their favorite thing to do is sit around and read. Note like TV is like huh? They're still on dial-up, they live in rural Pennsylvania and they're wicked, wicked smart.

Richard Campbell: And this is the Ben Franklin line you're talking about, right?

Carl Franklin: Yeah, yeah, exactly. We're descendants, second cousin to Benjamin Franklin.

Richard Campbell: Wow.

Carl Franklin: Joseph.

Richard Campbell: Intellectual rednecks.

Carl Franklin: Intellectual rednecks, yeah. It's really, really interesting.

Richard Campbell: Okay.

Carl Franklin: Anyway, that's all I'm going to say on that because otherwise we'll bore the listeners. Let's get into Better Know a Framework.

[Music]

Richard Campbell: You're right.

Carl Franklin: Better Know a Framework is this little section where I shine a little light on a dark corner of the .NET Framework, and oh boy, what a dark corner we have today.

Richard Campbell: Yeah, what have you found?

Carl Franklin: I found the RegCode Namespace...

Richard Campbell: Uh-oh.

Carl Franklin: Not System RegCode, not Microsoft.Windows.RegCode but just the RegCode Namespace, and there are three classes in here. There's the RegCode class, there's a Remote RegistrationClass, and you know the COM Object Registration Tool, Regasm? Well, this is basically a class for invoking that.

Richard Campbell: Oh, interesting.

Carl Franklin: Yeah.

Richard Campbell: So from .NET,

Carl Franklin: Yeah, from .NET. Now when you go to the Add References in Visual Studio, you won't find it in the list. I just did a little bit of searching here and found out that it's in Framework 1.1.

Richard Campbell: Only?

Carl Franklin: Only. I couldn't find it in 2.0, couldn't find it in 3.5, ODB in 2.0 otherwise, but it's in 1.1. I mean, that's all I could find out just looking very quickly for it but maybe an alert listener has some other information they could share with us about



Kiciman and Friedman Bring AjaxView to Studio!

July 27, 2009

RegCode, but that's what it's for. It's for using the Assembly Registration Tool, Regasm.

Richard Campbell: Interesting.

Carl Franklin: That reads the metadata in assembly and that's the necessary entry to the registry and that enables COM clients to create .NET Framework classes transparently.

Richard Campbell: Cool, wow. I'd love to get an email from someone talking about how they're using that. It's worth a mug too, that's for sure.

Carl Franklin: Sure, absolutely. If you've got emails like the one Richard is about to read, send it to dotnetrocks@franklins.net. Lay it on me, Richard.

Richard Campbell: Here we go. "Hello Carl and Richard. I've been working at a university for several years doing contract work for the city of Albuquerque where I'm located. I just heard on show 462, Paul Stubbs on Silverlight in SharePoint, that in Microsoft they're starting a new website to help developers in these tough economic times. Even my seemingly safe job isn't as secure as it was in the past so I'm interested in updating my skill set as you never know when you may need it. I'll be looking forward to what's on Thrive For Developers and the content that you both will be putting up there and I love .NET Rocks! I've been listening for the last couple of years. It's really the highlight of my week to hear both episodes. Please keep it up. Rod Fellanga." Thanks, Rod.

Carl Franklin: Yeah.

Richard Campbell: The site that Rod is talking about is called Thrive and its at msdnevents.com/ThriveDev and that's where our development in a downturn podcast series is being published every week.

Carl Franklin: Yeah. We did 10 shows and they're all about what you can do if you find yourself out of work or if you want to prevent yourself from being out of work, you know, how to survive it.

Richard Campbell: We've really got an interesting cross-section of viewpoints there and I think it's three or four shows published now but they're coming out every week so you can go back and listen to them all.

Carl Franklin: And that brings us to today's guests, Emre Kiciman and Mark Friedman. Emre is a researcher in the Internet Services Research Center, the ISRC, at Microsoft Research in Redmond where his interests are broadly in the area of large-scale Internet services, their operations, and their end-to-end reliability. Often, his work focuses on monitoring

and machine learning analysis of system behavior to improve reliability and performance. Most recently, his research has focused on web applications. Mark Friedman is our other guest. He is an architect on the Visual Studio Diagnostics Team. His main area of focus is the Microsoft Performance Tools including the Performance Profiler. He's been with Microsoft for three years and has written several books on performance. Emre, Mark, welcome.

Emre Kiciman: Nice to be here.

Mark Friedman: Thanks, guys.

Carl Franklin: Nice to have you, and Emre, nice to have you back.

Emre Kiciman: I'm happy to be back. Thanks for the invitation.

Carl Franklin: Yeah. In your last show, we were talking about AjaxView back in March 2008.

Emre Kiciman: That's right.

Carl Franklin: Show 324.

Emre Kiciman: So to give you a quick reminder or review of what AjaxView was, it's a research project that attempted to give web application developers better visibility into how the JavaScript code is running inside the end-user's browsers.

Carl Franklin: Yeah.

Emre Kiciman: So the idea is that you're writing these huge web applications these days, lots of them, and of thousands or hundreds of thousands of lines of JavaScript code. You're shipping them down to your end-users browsers. They're running it but you have no idea, the developer, whether your users are getting a good experience, whether it's fast or slow or even whether there's no exceptions. So AjaxView attempts to give that visibility to web app developers by automatically instrumenting the JavaScript code as it's sent out to the end-user's browser, and then basically the research was around how to make sure that that instrumentation is efficient and your users don't actually see any performance penalty because of the instrumentation.

Carl Franklin: Yeah.

Emre Kiciman: So now, what happened since we had the show is we've been working together with Visual Studio and we wish someday to release the Power Tool version of that and we can talk about that.



Carl Franklin: Sure, why not. Let's talk about it, a Power Tool.

Emre Kiciman: Okay. So what we did was we basically took our previous implementation, our research project, and it had been implemented as a proxy so we as researchers could experiment with any website we wanted. We put the proxy between the web browser and any server we want and we get to automatically instrument that JavaScript code. What we did in Power Tool is we've made it easier to use it for your own web server. So what we do is we took out the instrumentation bit and we plugged it into IIS so it's now a module that you just install in your web server and now all of your jobs that get served out can be instrumented. There's a little web admin interface, you check a box and you start instrumenting and collecting data and then you can download the profile for any of your JavaScript code into Visual Studio and explore it using Visual Studio Performance Profiler.

Carl Franklin: Very cool.

Emre Kiciman: We're very excited about it.

Carl Franklin: Okay and so that brings us to Visual Studio integration. So what's the story there?

Mark Friedman: The story there is that with Emre's help we've developed a Power Tool that's available out on Code Gallery that people can go and download. This Power Tool then integrates into Visual Studio. It's got a couple of different components and it integrates into Visual Studio 2008 at the moment and we are working to get it to sync up with Visual Studio Dev 10 and Emre's team, I think, is doing that and we're hoping to get that out soon and we're looking at getting feedback from customers to see where we should go with this. I mean, we have some broad ideas about how to do, I would say, a full job of end-to-end performance monitoring for .NET applications where we've got client pieces, we've got the server-side pieces, we've got the data tier, we've got web services tier, perhaps we've got the Cloud and what we are looking on the performance tools end is how to build a real holistic view of all that and then give you specific tools to drill into your specific problems.

Richard Campbell: So are we talking about -- this is just a debugging tool or do we live on it all the time in production apps?

Emre Kiciman: This is Emre again. What the Power Tool is is right now it's intended for testing environments. So it works at small scale. We haven't done all the engineering work so that it can work on a production site yet. That's obviously the direction we want to go but right now you can take the Power Tool,

install it on your IIS server, in your testing environment, run your test workload against it and use that data to give back to your developers information about the performance and everything.

Mark Friedman: The key technology that's in the AjaxView, Ajax Scope – what are we calling it these days, Ajax Scope?

Emre Kiciman: Sure, or that might be.

Mark Friedman: Something like that.

Emre Kiciman: Yeah.

Mark Friedman: Technology. Well, there's two major pieces of them. One of the keys is that there's an instrumentation piece where the JavaScript, just before it's downloaded to your web client, is instrumented and that instrumentation is what we're using to measure performance and that's injected into your app by a Server-side Component that's integrated with IIS and that's the way the instrumented app gets out in the field. Now we don't have a whole suite of management around managing that in basically a production environment yet, but that would be the general direction where we see this going. So we are recommending that people use this more in a test environment but the profiles that it produces basically tell you for your test scenario whatever your area of mode test kind of environment, how long functions on the client are taking, and nice integration into Visual Studio app.

Emre Kiciman: Of course, because of the JavaScript instrumentation that's being done on the server and it's all standard JavaScript, this is stuff that works across web browsers so you can be using the same Performance Profiling Tool whether you're running your test in Firefox or IE or Safari or even I think it works on Google Chrome.

Carl Franklin: Wow.

Mark Friedman: And transparent, that's where we've seen that one of the issues in getting a tool into the browser and for us the browser that we could put tools like this into, of course, would be IE but you've got to get customers opt in on that and then things are being loaded down and the browser is filled with all kinds of security these days to keep extensions from occurring.

Richard Campbell: Right.

Mark Friedman: The technique that Emre and his folks here have pioneered no need to worry about any of that. You basically download an instrumented version of the Script.



Richard Campbell: Yeah. I know my dev machine's browsers are full of HTP watch and Fiddler and Firebug and all of those kinds of things, but my clients don't have any of that stuff. You know, we never have these performance problems in our own machines.

Mark Friedman: Yeah and you know, people are starting to really develop, specially with the libraries that we have now that let you do Ajax, they're starting to get some fairly heavyweight development at the client and we're scrambling on the tool-side to catch up to them as we make things like Web Services easier and easier to get to from the client-side; we've got to get the tools into place that let people evaluate what's really going on. So now it's performance oriented, it's not designed to be a debugger but we see some potential applications in the basic technology for things like Code Coverage and Test Impact Analysis which are also things that are coming for mainline applications in dev 10.

Richard Campbell: So what kind of info are we getting out of this, like what is it telling us about the AJAX running on the browser?

Emre Kiciman: What the power Tool captures right now is the performance profiles of each of the functions that are defined in your own script.

Richard Campbell: Okay.

Emre Kiciman: So essentially what we're doing is we're putting a start and an end bugging function into every function that you write and then we capture that data. So it gives us performance profiles, as well as, call traces and then you get to aggregate that up and see where the problems are in your code from the performance point of view. There's obviously extensions you could do to capture more data for debugging and that's something we're looking at.

Mark Friedman: So it would be equivalent to, among the profiler options, would be equivalent to an instrumented profile because there's code added to the prologue and epilogue of every function call. You could see that just like with the instrumentation, we could see options which aren't there now. We could extensions where you'd have some more selectivity about what functions you want to instrument, but for instance if your client-side code is spending a lot of time in a Web Service call, we would have that but of course that is also asynchronous that we've been talking about how do we make that kind of response time oriented, but anyway, we've got the basic measurements and now we're thinking about how to really make this stick to the application model that you're using.

Richard Campbell: Does it instrument everything? Like I'm merely thinking that I'm surprised how many times I've found out that my performance problems were caused by something like a Google Analytics bug added into my web page. Will it pick that up for me too?

Emre Kiciman: Because it's a server-side integrated instrumentation, it is only able to instrument things that come from your IIS server.

Richard Campbell: Right.

Emre Kiciman: Yeah. So in that case, I think if the Google Analytics code is causing problems, it's being served from Google's own TBN or so, you'd have a bit of difficulty capturing that unless it was some third party code that you're calling into because then we build the capture from your code the fact that that was taking a long time.

Mark Friedman: But that's definitely one of the challenges. We're trying to figure out how to take this tool and integrate with it further knowledge of what's going on. Complimentary to something like the AjaxView Power Tool is another tool that's available called VRTA, Visual Round Trip Analyze, that sits on top of the network monitor and that can give you insight into basically a page load time and where your delays are around specific. You know, you'll see in this pages something getting whack in into your page from an outside server that's serving up an add or analytics or all of those pieces show up in the broader trace, but that's not visible necessarily from the JavaScript. One of the things that's interesting about the JavaScript though, as I said, is that's definitely the model that people are using the program for web apps these days so we're seeing more and more complexity there and those complex scripts are very difficult to categorize in terms of their performance so this is a technique to get it back, a piece of it, but we're still just getting pieces and our challenge for the tool is to get you that whole picture of what's going on at every level of your app. It's a challenge because we can't stop the other team from Microsoft from releasing new stuff all the time and we do scramble on the tool side just to keep up with them.

Richard Campbell: Sure, yeah. That could be very challenging, there's just so many moving parts going on here trying to avoid breaking things but I guess I'm trying to get the key points here that you add in the profiling code at the server level and you're basically just identifying your marking methods in JavaScript.

Emre Kiciman: That's right and then the client captures all these messages about, you know, I entered function foo at times zero and then I exit at times two, and it aggregates all that up and then



pushes it back to the web server so the developer can then download and take a look at those profiles.

Carl Franklin: How does this compare with something like Firebug?

Emre Kiciman: Firebug is a great tool if you're debugging your own applications from your computer.

Carl Franklin: Okay.

Emre Kiciman: But if you want to get visibility into how your JavaScript is running across lots of different people's machines, across lots of different browsers, then you need a tool like our AJAX Profiling, our tool, that will be able to, in some of the JavaScript itself, so that you can work across all these environment.

Carl Franklin: Firebug is really a one client, one person at a time kind of tool.

Emre Kiciman: Yes.

Carl Franklin: In other words, yeah.

Richard Campbell: And really that's the tool that the dev runs. Here you're actually shipping code to the client, to the user.

Carl Franklin: Yeah, yeah.

Emre Kiciman: That's right and we would definitely want to move into like looking forward into where directions for this Power Tool. We'd love to move this into a production environment where you're actually typing real world data and you get to see what your real users are seeing and that's really very exciting because that's great data. Performance really affects how much time you will spend on your site, what they do there, whether they stay engaged once they distracted and go off and do something else. Waiting for the performance is really critical with these web apps and we'll see how planning goes going forward, but obviously that's a great direction.

Carl Franklin: This portion of .NET Rocks! is brought to you by our good friends at Telerik without whom this show would not exist. No doubt you bumped into testing tasks now and then in your work and we can bet writing functional test is not your favorite thing. It's difficult. It takes ages and the results could be dubious. Well, get ready to start liking it, thanks to Telerik. With the just launched WebAii testing framework, building web automation tests is a breeze. Enjoy codebase automation of advance ASP.NET AJAX and Silverlight apps, write a single test and have it executed against multiple browsers at once, benefit from rich API LINQ support, integration Visual Studio unit testing NUnit, xUnit, and

MbUnit, not to mention the free wrappers for a Telerik RadControl for ASP.NET AJAX and Silverlight as shipped with Telerik's new testing tool, surely one of its best features. WebAii testing framework, which is developed by ArtOfTest, is absolutely free. If you're already hooked on WebAii testing framework, you can start using it right away. Go to www.telerik.com for more info, and hey, make sure you thank them for supporting .NET Rocks!

Richard Campbell: So guys, how are you sending this data back to those servers? Is this all like Web Services calls? AJAX calls?

Emre Kiciman: Yeah. So we take the logged data. We don't talk to the client, we just talk to a list in JavaScript basically just keep appending every log message that we generate about these time stamps and performance information. It took configuration every five minutes or 60 seconds or 30 seconds or whatever you want to set, that client-type code triggers a timer and then that timer grabs his list of stuff, packages it into an XML, HTTP request over into the server basically posting it up to your IIS.

Richard Campbell: Right. I'm worrying about performance impact here as to how much overhead am I adding by doing this, but the fact that you're not sending it every time a method call is happening makes me happy, you know. But rather consume a memory first then ship it periodically, and then you can dial it down if you're worried?

Emre Kiciman: Inside our instrumentation, there's a config that's easy for, I guess it's easy for us to change. I'm not sure if it's exposed to use as a Power Tool but if there are problems with that, the people are seeing if that's the design, obviously we'd love to hear this type of feedback.

Richard Campbell: Okay.

Mark Friedman: We don't have a lot of controls around managing the volume of data that can be generated across the large number of users or a pretty complicated website. So that's the piece that obviously needs to be something we're going to move forward with if people like it and think it's the right direction for us to be going, but we wanted to make it available. We thought at this point it's more than usable in having visibility into the JavaScript-side, the client-side of your AJAX application. I don't know how else you can get it at the moment.

Richard Campbell: Right.

Mark Friedman: We do have some other things that are shipping in Visual Studio 2010 that rely on a specific version of internet Explorer and some APIs



that are there, and we can do that sort of thing too but that is depending on you having IE on the client.

Richard Campbell: Yeah. Then you're back to browser specificity and that causes its own set of problems.

Emre Kiciman: I love to jump in here and tell you a little bit about the things we do to minimize the performance impact on the research-side of things, the stuff that isn't in the Power Tool yet but the techniques that we have explored and we know work.

Richard Campbell: Maybe in a coming version?

Emre Kiciman: In a coming version, sure.

Carl Franklin: Yeah, so what's...?

Emre Kiciman: So what we do is we explore two techniques. One is the distributed instrumentation, and one is adaptive instrumentation. The distributed instrumentation is really simple to explain. We basically take a look at all the different places we could be adding instrumentation to a program and we say every user is going to instrument maybe a percent of these things. So now you have thousands of users to your website and each of them comes in and gathers a little bit of data for you and sends it up to the server. But in aggregate, the developer gets complete view of the performance information.

Richard Campbell: Oh, I see, yes. So rather than logging every method on every browser, you log a subset of them and the net is what you wanted anyway.

Emre Kiciman: Exactly, exactly and this means that no one is running all the instrumentation code so no one sees that potentially slow version. Everyone is just running a small set, everyone is happy and all the data gets collected in the end. The second one, adaptive instrumentation, that one is a technique that actually looks at the incoming performance profile data and users have to figure out what should be instrumented next. So if a user comes in, downloads something and you find out the MouseDown Event Handler is really slow, well, then, the adaptive kind of engine in the instrumentation would say what functions are getting called inside the MouseDown Event Handler, let's drill down and instrument those then we'll see which of those are slow and we'll then instrument another level and another level and basically drill down until you get to the real problem in your code and this works because in the web app environment you always have new users coming in downloading new versions of the code so you can basically get very quick turnaround on data for a new kind of instrumentation that you want to do.

Mark Friedman: So you can see by a combination of those two techniques that you can start to build something that really can handle production volumes which is where we want to go with this kind of tool.

Richard Campbell: Yeah, if you're running a site doing a hundred transactions a second, you're going to be able to drill through fairly quickly to download the thing that actually matters without trying to get everything in one request.

Emre Kiciman: That's right, that's right. There's obviously some kind of aggregation you have to make sure you do to resolve variances in people's hardware and things like that. You're going to have to take an average of a lot of users to get good data but if you have hundreds of transactions going on, yeah, you're going to get there pretty quick.

Mark Friedman: One of the things we're leveraging in the Power Tool that's interesting is new features of IIS, basically something called the integrated pipeline. Have you guys done shows on that in the past?

Carl Franklin: Yeah.

Richard Campbell: I think we've done a show -- I'm aware of it, but I don't know that we've done a show specifically around this. This is part of IIS 7.0.

Mark Friedman: Part of IIS 7.0 and there's now a very explicit architecture that any web application is executed, in specific what they call pipeline stages, and this allows us to plug-in a module into the pipeline. So we have a very straightforward way of hooking into IIS at the time, at the point where your JavaScript is being downloaded and getting before and after, putting the instrumentation in and then shipping it out, and that integration with IIS is also critical because we have an IIS component that's then responsible for -- or managing when you have somebody to send it to back at the client level. So getting this into IIS7 was a pretty cool thing.

Richard Campbell: Just from a perspective of being able to be very precise for the details. All feels very much like regular .NET profiling, again, you'll notice doing all those tricks of JavaScript. Is the view client the way that I look at this data? Is this all in Studio then? There's no other way to look at it?

Mark Friedman: It absolutely looks exactly like the same experience you get with the Visual Studio profiler.

Richard Campbell: Okay.



Mark Friedman: It's just a different kind of profiler file that you open up.

Richard Campbell: Yeah, you open it up from Studio, because I'm just thinking in long term this is running fulltime. You probably wouldn't want to look at this in the studio; you'd want some other client.

Mark Friedman: We do and the kind of thing that Emre is talking about in terms of being adaptive, I think I've been experimenting with the idea of, well, let's make this service level oriented. Maybe we have some very high level measurements that measure service levels and then only start to inject more and more instrumentation into your code adaptively when there are problems, and in the web environment that would work as Emre said because you're constantly introducing new users into the mix and new opportunities to download the code. So yeah, yeah, once you have brought coverage, once you're looking at -- we have to create the data, I'm not sure Visual Studio then becomes the right way to display this data. We might create the data and then open it up for different management framework, for instance.

Richard Campbell: I'm also starting to think in terms of the same way we do with like SilverLight's ability to dynamically adjust the quality of the video based on your connection; that you could start to adapt being your app based on the strength or performance of the browser at the other end.

Carl Franklin: Yeah, adaptive streaming.

Mark Friedman: Yeah. Don't you think, I mean, most of the time the performance is I just need to know reporting and I just need to know that I'm hitting my goals and 99% of the time I am in and of course 99% of them are clients. So clearly you don't want to instrument everything. You want to be able to instrument intelligently and, yeah, I think that's the way to go.

Richard Campbell: I'm also thinking in terms of I'm going to give this guy a lighter weight site because his machine can obviously not hack it.

Emre Kiciman: That's a very interesting idea and if the web developer is willing to basically code up versions or mark certain features as being heavyweight features, that definitely is something that you could determine based on this profile data.

Richard Campbell: Yeah.

Emre Kiciman: One other project that we have been working on, again on the research-side of things, is a project that takes this performance profile data and uses it to figure out what code runs early in the initialization of the page and the basic features

and what code users tend to not to call until much later, or perhaps not to even call at all, and what we do is we automatically rewrite the JavaScript code so that you only download the code that you need initially and then all the extra features, everything gets downloaded in the background. So essentially what we're doing is we're stopping out the functions you're not going to use, shrinking them down to like I think then we get down to 20 to 40 bytes for a function, and then if that function is needed basically it will be downloaded later either in the background or pushed down at the time that you need to call it, just in time kind of delivery of your functions, yeah, or delay loading or all sorts of, yeah.

Carl Franklin: Richard, your app server from...

Richard Campbell: Strangeloop, yeah.

Carl Franklin: Yeah, Strangeloop.

Richard Campbell: AS1000.

Carl Franklin: Yeah, the AS1000 does some really interesting stuff with images. Maybe you could talk about that.

Richard Campbell: I was thinking specifically we have that feature in our appliance for identifying JavaScript chunks that aren't used as part of the load, part of the page.

Carl Franklin: Yeah.

Richard Campbell: We move them down so that their asynchronously loaded at the end of the page. Now you get into all kinds of performance things that we get into spriting images together to minimize round trips and so on. That's all about creating performance opportunities, but it's interesting to see how you guys are getting into that same sort of place where by having this profile data you know what methods are being used and not used. You've got a really interesting insight into what could be asynchronously loaded late and what needs to be available early.

Mark Friedman: Yeah, again as the programs get more and more complicated, it's not really easy to do a kind of a static analysis or just a code. You really need to understand what data is being access, what passed through the code that uses the application or executing and yeah, the only way I think to really do it effectively is to be monitoring it as you go.

Richard Campbell: There's no substitute for empirical data. What did they call?



Emre Kiciman: You bet, you bet, yeah and if people want to read more about this project and we can talk about more of your questions too but I just want to make sure we give the reference. This is a joint work with colleague Ben Livshits at MSR and the project name is DOLOTO. That stands for, depending on who you asked or when you ask us, actually the Download Time Optimizer or it's Russian for chisel, chipping your way through JavaScript.

Mark Friedman: And then we'll no doubt appreciate that plug, can't we. I'm sure you will.

Richard Campbell: When we were originally talking about doing a show, I was fascinated to see here is some Microsoft research project that's becoming a shipping product, a part of the suite, but there's still a research part of this going on.

Emre Kiciman: Oh yeah, there's so much data here and the web environment is really very exciting in terms of having service-type control over what your users are running. Whenever they come in, you have another chance to decide and give them one thing or the other in terms of running AB tests around what actually helps performance or not, in terms of really being able to get detailed instrumentation. Yeah, it's very exciting.

Mark Friedman: Well, on the product development side, we are just thrilled with the work that Emre and his folks over here have done, and their insights into the architecture, of browser, and JavaScript were quite frankly skills that we didn't have internally so we were very glad to pick up on their work and encourage it and work with them.

Richard Campbell: How different is the mindset of a researcher to a guy who builds products? Like did you find...?

Emre Kiciman: We're looking at each other right now.

Richard Campbell: Yeah. I don't mean to put you guys on the spot because there you are, but I got to think that...

Mark Friedman: It's a little different. It's a little different...

Richard Campbell: Yeah, it's got to be different.

Mark Friedman: From what we found working together, it's our deliverables are usually a little different but here we were able sync up when getting the same deliverable and the same call. For us in the product side, we don't care until we ship it and it's great to have all these great ideas and you know, in a way I could get great ideas or dive some out because

we all have them. Can we implement, can we put all the quality, can we clear all the quality gates around it, create the documentation, create the whole experience; we have to do that before we're ready to ship it. So yeah, that's a little different. But in terms of the ideas, sometimes we're driving something that look pretty "researchy" and sometimes these guys over at MSR are doing things that look eminently practical. How about that?

Richard Campbell: So I guess the other layer of this question will be how much of the codebase that was written by MSR actually made it into the product.

Emre Kiciman: For the Power Tool actually good chunks of it. We did swap out the JavaScript parts that we were using but we kept all the instrumentation code, things that analyze the code and figure out where to put instrumentation and I think even though it's not exposed a lot of the logic for doing the adaptive and distributive instrumentation is still there although it's stopping call by Power Tool, that probably didn't count.

Mark Friedman: And then on our side, what we did on the product-side, we put this group, the ringer, in terms of quality testing and that it support all the versions of the browser that we needed, all the versions of IIS, etc. We cut this in flight. When did we release this? Probably about March?

Emre Kiciman: I think end of April.

Mark Friedman: End of April. So Dev 10 was on its separate development app so we decided not to sync up at that point but we're fully ready to sync up in the future and get this integrated into the Dev 10 build and we have a lot of new features on the profiling side that really we should make this very exciting addition to what people are getting.

Richard Campbell: There's always an interesting dynamic to profiling and I've played with a lot of different tools about helping you find real problems. It's not just what run long but what's used a lot so that sort of combine -- you know, if this may not be the slowest thing but it's used so much compared to that slower thing, then it's probably more of a concern.

Mark Friedman: I like what you said about the empirical observation.

Richard Campbell: Yeah.

Mark Friedman: Often if you ask the developer, they might give you an answer based on, oh, I don't know, the part of the code that they wiggled the most, or the part of the code that they found most challenging, or something like that. But the empirical way of looking at it gives you, in a much more



grounded way, this is the actual bottleneck, this is the data that I'm getting. So the tools play a real role there and it's a good idea to get from the anecdotal evidence to the empirical evidence.

Richard Campbell: Where are you guys on capturing errors at the JavaScript level on the browsers?

Emre Kiciman: Right now, I believe the -- let me think. The Power Tool does the -- client-side instrumentation does capture exceptions and sends back the messages and together with the function, you know, the trace of the functions that you've call and you can get that whole stack trace and everything. We've done some research into also capturing things like local variables and stack and everything but that's not in there. But I believe what happens with this data right now, since the performance profiler doesn't show the exception, it's that the data just logged into file but not exposed. We have the interface.

Carl Franklin: So you can go look at it if you want to.

Emre Kiciman: Yeah, you could open up the file with your text editor and put through. I wouldn't say that that's a great experience.

Richard Campbell: Yeah, but you know, that would be the thing I'd want on for everybody. If something in my app is throwing exceptions on the client, normally I would never see them and most customers are not capable of reporting that effectively, getting that back to my server I would add that to my log analysis.

Mark Friedman: That's a really interesting idea. We do have some of that data today, and that whole area of, well, we thought in general that the technology -- I mean, the diagnostic team where we make the debugger, the newest historical debugger, things of that sort and again in getting a better experience out of this client-side code, it's one of our interest including test impact analysis and stuff like that. So yeah, one of the really critical aspects of this technology is the interface that we have between the client and the IIS server and we aim to exploit that going forward.

Emre Kiciman: It is funny. Getting visibility into the errors that occur on the client-side also leads to some interesting challenges. One of the things that we found, for example, and this is an anecdote that I really delight, is that when the browser reports an exception, it does it in the local language. So if you are actually capturing this data on your server-side, you then get the same error message translated into German and Japanese and English and Spanish and everything and you're going to have to basically flip

through there to figure out which ones are the same and which ones are not and all basic things like the call stack trace and everything can help there, but it's a funny thing about what happens when you start tracing errors that happens in the client-side and not intended for the original web servers, the developer, and you start exposing that over.

Richard Campbell: The other thing that's going to happen is you're going to get a new browser deployed and it's going to behave differently on something and that's the first warning that you're going to have problems with that browser as these messages are coming back.

Emre Kiciman: Yes. So that's actually a great analysis to take a look at, is to take all your incoming performance profile and bracket them by user agent and see how they're different.

Carl Franklin: Yeah.

Richard Campbell: You sort of get a feel for how the different JavaScript stacks actually execute on those different browsers.

Emre Kiciman: That's right and you'll also see, I'm sure, biases because newer browsers are likely to be deployed on better hardware...

Richard Campbell: Right.

Emre Kiciman: You know, your older versions of browsing on the slower hardware and things like that.

Mark Friedman: Well, and it's the same story where the browser, it's the browser where's today about who is running faster, they're running different benchmarks applications. Are those applications your applications? That when this kind of soar you can actually quantify that very well and compare the performance across different browsers and there you go. What matters is your workload, not these pie in the sky benchmarks that don't have anything to do with what you're trying to accomplish.

Carl Franklin: So is this tool an add-on that we can download a CTP of? Is there anything available now?

Emre Kiciman: Of course, yeah. The name of the tool is Microsoft Visual Studio AJAX Profiling Extension Power Tool.

Mark Friedman: And we had the marketing people come up with that sexy name.

Carl Franklin: That's awesome.



Kiciman and Friedman Bring AjaxView to Studio!

July 27, 2009

Emre Kiciman: Yeah. So we still call it AjaxView, but the official name is AJAX Profiling Extension. The URL is code.msdn.com/AjaxView.

Carl Franklin: Would you mind if we called it the AjaxView Power Tool?

Mark Friedman: Not a bit. We wouldn't mind, not a bit. But it does require Visual Studio Team System or Visual Studio Developer Edition. It does require the profiling of course.

Emre Kiciman: Yeah, yeah and I think it also requires -- some of the APIs we use were only introduced in SP1.

Richard Campbell: So you do need the Service Pack 1.0 of Studio installed.

Emre Kiciman: Yes, correct.

Richard Campbell: This is 2008 we're talking about right now.

Emre Kiciman: That is right, 2008.

Richard Campbell: And then on the back-end, it's IIS7?

Emre Kiciman: IIS7 on the back-end and we can then profile your applications as long as you have the integrated pipeline turned on.

Mark Friedman: At the website that Emre just gave you, there's a set of requirements that you can look at and verify that you match up to those, there are a couple of documents that will help you get started, and I think there are pointers to the research that MRA has collaborated that they have done where people can kind of look at what the thought behind this was and where they think it needs to go.

Carl Franklin: Excellent.

Richard Campbell: But development is ongoing. This is only version 1.0. Obviously we've talked about some of the new stuff you guys have got coming as well.

Mark Friedman: We're looking at this to sync this up to Dev 10 so that we have an integrated Power Tool there. Emre's work was done though in kind of an isolation really from what we were doing with Dev 10 so really how we go forward with this is a Dev 11 issue for us and quite frankly right now I'm not thinking about Dev 11. I'm heads down trying to finish Dev 10 and getting a quality release out there but you'll see in Dev 10 there are a number of functions, profiling functions that we've had and we could go into them at a later time, that will also give you an

enhanced view of your application and I think people are going to like them as well. This is a very nice piece that we're happy to add.

Richard Campbell: Yeah. Debugging has been a huge feature add for Dev 10 but this, I guess, is coming a little too late to make Dev 10 per se?

Mark Friedman: Well, it's a Power Tool so it's kind of what we operate. We operate one big ship train around Dev 10, and the Power Tool will actually have the advantage of doing it that way so we can ship separately and we don't have to sync up and we can, as it's available, make it available.

Richard Campbell: Right.

Mark Friedman: So we're hoping to sync-up around the beta 2.0 of Dev 10 and then have a Power Tool version that then you can plug in.

Richard Campbell: Yeah, there's advantage just being independent like than rather being in a big ship.

Mark Friedman: It really is and we're looking at ways at being a little bit more agile about what we can ship and how we can do it especially as the profiler set of tools get to be much, much more diverse and quite frankly we ought to be able to ship more and more of them on a faster schedule and we'd like to do that.

Richard Campbell: Yup. I suddenly put my IT hat on and saying, wow, you know, I really like it if you fed data to System Center Operations Manager too.

Mark Friedman: Well, one of the features that we're doing with Dev 10 inside the historical debugger is we've got that integration going with Systems Center for the first time.

Richard Campbell: Oh really.

Mark Friedman: Yeah, yeah, so now the fact that something failed is forward to the Systems Center and you can do alerting around it and we're looking at that as the initial piece of that and as we go forward in the whole diagnostic and performance space, we're looking at emphasizing that more and more. As I said, we would look at this kind of data, this performance data, sort of at least service level data as being making available to any of the providers who are going to build management tools but Systems Center is right up there and, you know, right down the street from us so that we'd work with them too.

Carl Franklin: Well, we're just about out of time. You guys have any last minute things you want to throw out there?



Emre Kiciman: I just want to encourage folks to take a look at our AJAX Profiling Tool or AjaxView Tool, you can call it what you want. We would love to get folks' feedback, what works well, what doesn't work just to give us a better idea of where people think that we should be going with this.

Carl Franklin: All right, Emre, thank you.

Richard Campbell: How do we get in touch of you, Emre?

Emre Kiciman: You can email me at emrek@microsoft.com, and also email at ajaxview@microsoft.com to get the whole team and of course all these, just contact and things are up on the Power Tool download site together with even a discussion forum that we monitor and respond to so anyone of these ways will lead you to us.

Carl Franklin: Excellent. I think you'll probably get some good feedbacks from our listeners.

Emre Kiciman: I'm sure I would, I'm looking forward to it.

Carl Franklin: Okay. Emre, thank you. Emre Kiciman and Mark Friedman have been our guests. Check out the website, give them some feedback, and we'll see you next time on .NET Rocks!

[Music]

Carl Franklin: .NET Rocks! is recorded and produced by PWOP Productions, providing professional audio, audio mastering, video, post production, and podcasting services, online at www.pwop.com. .NET Rocks! is a production of Franklins.NET, training developers to work smarter and offering custom onsite classes in Microsoft development technology with expert developers, online at www.franklins.net. For more .NET Rocks! episodes and to subscribe to the podcast feeds, go to our website at www.dotnetrocks.com.