

# Great Lakes Software Symposium

Westin Chicago Northwest

November 13 - 15, 2009

<http://www.nofluffjuststuff.com/conference/chicago/2009/11/home>

Fri, Nov. 13, 2009					
	Salon 3-5	Salon 1-2	Chambers	Gallery	Barrington
12:00 - 1:00 PM	REGISTRATION				
1:00 - 1:15 PM	WELCOME				
1:15 - 2:45 PM	Effective Concurrent Java Brian Goetz	Common AntiPatterns and How To Avoid Them Mark Richards	The Busy Java Developer's Guide to Java Platform Security Ted Neward	Groovy for Java Programmers Venkat Subramaniam	Preproduction: Everything You Need to Do Before Iteration 1 David Hussman
2:45 - 3:15 PM	BREAK				
3:15 - 4:45 PM	The Java Memory Model Brian Goetz	On Being a Software Architect Mark Richards	The Busy Java Developer's Guide to Advanced Platform Security Ted Neward	Cleaning up Code Smell Venkat Subramaniam	Producing Software Groove David Hussman
4:45 - 5:00 PM	BREAK				
5:00 - 6:30 PM	Are All Web Applications Broken? Brian Goetz	Transaction Pitfalls and Strategies Mark Richards	The Busy Java Developer's Guide to ClassLoaders Ted Neward	Testing with dependencies Venkat Subramaniam	Discovering Real Value with Story Maps and Personas David Hussman
6:30 - 7:15 PM	DINNER				
7:15 - 8:00 PM	Keynote: by Ted Neward				

Sat, Nov. 14, 2009					
	Salon 3-5	Salon 1-2	Chambers	Gallery	Barrington
8:00 - 9:00 AM	BREAKFAST				
9:00 - 10:30 AM	Open Source Debugging Tools for Java Matthew McCullough	Real-world JEE performance tuning: Tips n' Tricks Pratik Patel	Stupid JIT Tricks Brian Goetz	Introduction to JMS Mark Richards	What Is Lean and Why Should You Care? David Hussman
10:30 - 11:00 AM	BREAK				
11:00 - 12:30 PM	Open Source Debugging Tools for Web Apps Matthew McCullough	Enterprise JPA & Spring 3.0 - Tips and Tricks for JEE5 Persistence Pratik Patel	This is not your father's Von Neumann Machine: A crash course in modern CPU architecture Brian Goetz	Advanced Topics in JMS Mark Richards	Connecting Companies with Acceptance Testing David Hussman
12:30 - 1:30 PM	LUNCH				
1:30 - 3:00 PM	Git Going with Distributed Version Control Matthew McCullough	JUnit4: Take your testing to the next level Pratik Patel	JSF 2.0: An Introduction David Geary	Spring and JMS: Message-Driven POJOs Mark Richards	Coaching and Producing Agility David Hussman
3:00 - 3:15 PM	BREAK				
3:15 - 4:45 PM	Mastering Maven 2.0 Matthew McCullough	Groovy and Grails in the Enterprise Pratik Patel	JSF 2.0: Advanced Topics David Geary	The Busy Java Developer's Guide to Hacking with the JDK Ted Neward	Architecture and Agility Are Not Enemies David Hussman
4:45 - 5:45 PM	BIRDS OF A FEATHER SESSION				

Sun, Nov. 15, 2009					
	Salon 3-5	Salon 1-2	Chambers	Gallery	Barrington
8:00 - 9:00 AM	BREAKFAST				
9:00 - 10:30 AM	Emergent Design & Evolutionary Architecture Neal Ford	Virtualization for development Pratik Patel	Programming Scala Venkat Subramaniam	Flex for Java Developers David Geary	REST : Web Architecture for Rich Clients Brian Sletten
10:30 - 11:00 AM	MORNING BREAK				
11:00 - 12:30 PM	Real-world Refactoring Neal Ford	The Busy Java Developer's Guide to Collections Ted Neward	Tackling Concurrency on the JVM Venkat Subramaniam	Cloud Computing Boot Camp on the Google App Engine Matthew McCullough	RDFS : Weaving Richness and Meaning in the Web Brian Sletten
12:30 - 1:15 PM	LUNCH				
1:15 - 2:15 PM	EXPERT PANEL DISCUSSION				
2:15 - 3:45 PM	The Productive Programmer: Mechanics Neal Ford	The Busy Java Developer's Guide to Functional Java Ted Neward	Building External DSLs Venkat Subramaniam	GWT fu, Part 1 David Geary	SPARQL: Querying the Data Web Brian Sletten
3:45 - 4:00 PM	BREAK				
4:00 - 5:30 PM	Communication Skills for Knowledge Workers Neal Ford	The Busy Java Developer's Guide to Advanced Collections Ted Neward	Effective Java Venkat Subramaniam	GWT fu, Part 2 David Geary	Semantic SOA : Meaningful Service Strategies Brian Sletten

# Great Lakes Software Symposium

Westin Chicago Northwest

November 13 - 15, 2009

<http://www.nofluffjuststuff.com/conference/chicago/2009/11/home>

## **Emergent Design & Evolutionary Architecture by Neal Ford**

Most of the software world has realized that BDUF (Big Design Up Front) doesn't work well in software. But lots of developers struggle with this notion when it applies to architecture and design. Surely you can't just start coding, right? You need some level of understanding before you can start work. This session describes the current thinking about emergent design & evolutionary architecture, including both proactive (test-driven development) and reactive (refactoring, composed method) approaches to discovering design. The goal of this talk is to provide nomenclature, strategies, and techniques for allowing design to emerge from projects as they proceed, keeping you code in sync with the problem domain.

## **Real-world Refactoring by Neal Ford**

Refactoring is a fine academic exercise in the perfect world, but we don't really live there. Even with the best intentions, projects build up technical debt and crufty bad things. This session covers refactoring in the real world, at both the atomic level (how to refactor towards composed method and the single level of abstraction principle) to larger project strategies for multi-day refactoring efforts. This talk provides practical strategies for real projects to effectively refactor your code.

## **The Productive Programmer: Mechanics by Neal Ford**

Developers from the 1980s would be shocked at how inefficiently developers use their computers because of the advent of graphical operating systems. This talk describes how to reclaim productivity afforded by intelligent use of command lines and other ways of accelerating your interaction with the computer and bending computers to do your bidding. Stop working so hard for your computer!

## **Communication Skills for Knowledge Workers by Neal Ford**

Software is fundamentally a communications game, and good skills differentiates between good and great developers. This session describes communication techniques and skills to people who skipped English 102 to hack some code. I talk about effective communication techniques for presentations, documentation, memos, and how to sell your technical ideas to a non-technical crowd.

## **JSF 2.0: An Introduction by David Geary**

This session introduces JSF 2.0 fundamentals, with emphasis on new features in JSF 2.0. **Prerequisite:** *Familiarity with JSF, or other component-based frameworks*

## **JSF 2.0: Advanced Topics by David Geary**

This session covers two of the most important features of JSF 2.0: composite components and built-in Ajax. **Prerequisite:** *Familiarity with JSF, or other component-based frameworks. Familiarity with Ajax. This session builds on the JSF 2.0 Introduction talk, so it is helpful, although not required, if you attend the intro talk before coming to this session.*

## **Flex for Java Developers by David Geary**

An introduction to Flex for Java developers. **Prerequisite:** *Familiarity with Flex and at least one other web application framework*

## **GWT fu, Part 1 by David Geary**

Learn to implement web applications with GWT. **Prerequisite:** *Familiarity with a component-based framework, preferably a desktop application framework*

## **GWT fu, Part 2 by David Geary**

Learn to do amazing stuff with GWT. **Prerequisite:** *GWT fu, Part 1 is not a prerequisite for this session, but it will help if you have some familiarity with GWT.*

## **Effective Concurrent Java by Brian Goetz**

The Java programming language has turned a generation of applications programmers into concurrent programmers through its direct support of multithreading. However, the Java concurrency primitives are

just that: primitive. From them you can build many concurrency utilities, but doing so takes great care as concurrent programming poses many traps for the unwary.

### **The Java Memory Model by Brian Goetz**

What's the worst thing that can happen when you fail to synchronize in a concurrent Java program? Its probably worse than you think -- modern shared-memory processors can do some pretty weird things when left to their own devices.

### **Are All Web Applications Broken? by Brian Goetz**

Many developers believe that web frameworks "take care of" the details of concurrency, but this is only because most web applications make limited use of state. Stateful web applications also need to be careful about hazards like races. This talk will use the Java Memory Model to analyze common patterns of state management in web applications. *Prerequisite: The Java Memory Model*

### **Stupid JIT Tricks by Brian Goetz**

Ever wondered what happens to your bytecodes when they're executed by a Java Virtual Machine? This talk provides a peek "under the hood" of modern JVMs, exploring dynamic compilation, speculative optimization, garbage collection, and some hardware-specific optimizations.

### **This is not your father's Von Neumann Machine: A crash course in modern CPU architecture by Brian Goetz**

Do software developers need to know anything about CPU architecture? They do if they aspire to be performance experts. Modern CPUs behave almost nothing like the sequential Von Neumann machine model we know and love. This session provides an overview of the architecture of modern CPUs, how this has changed in recent years, and what the implications are for software development and performance management.

### **Preproduction: Everything You Need to Do Before Iteration 1 by David Hussman**

Many agilists take little time to prepare for the first planning session of their first iteration on a new project. They dive right into the "work" and, sometimes, ultimately deliver software that lacks much value. Some newly formed teams believe that collocation breeds instant success and altogether ignore early planning. While sitting together always helps, it does not mean that people spontaneously collaborate to create sustainable value. Before holding the first planning session, preproduction helps communities learn about each other, the value they will deliver, and their newly forming ecosystem.

### **Producing Software Groove by David Hussman**

Agility comes in many forms. While you may start out using XP or Scrum, long term success will mean finding a groove which fits your company. This session provides a path for adopting or adapting agility which draws on the strength of the successful practices being used.

### **Discovering Real Value with Story Maps and Personas by David Hussman**

While actors and use cases often left users behind, personas and story maps bring the users to life and help mine real value. This session will teach you how to craft personas and use them to drive value into your development stream. The tools presented will help you better understand your buyers and users and build strong product backlogs and product road maps.

### **What Is Lean and Why Should You Care? by David Hussman**

Whether it was intentional or not, the agile community has been borrowing successful ideas from the lean manufacturing for years. Lean practices, like finding and removing wasteful work, can be applied without needing special permission or certification. Ideas like kanban (visual planning aids) and kaizen (continuous learning) are simple, helpful tools that are easily applied and produce great results.

### **Connecting Companies with Acceptance Testing by David Hussman**

While more companies are not waiting to test, testing is still something that is too often owned by the testers. This session provides tools and techniques which draw on the value automated testing provides while also using it as a core tool to help connect community. From idea to implementation, we will discuss and practice

various ways to connect business, development and testing around the value captured in the acceptance tests and their use.

### **Coaching and Producing Agility by David Hussman**

This session speaks to the question being asked so often: "what is an agile coach" From guiding planning sessions to pairing with developers to helping teams groom backlogs, this session provides insight into coaching agile projects and fostering the agility that produces. Whether you are a manager, developer, or tester, if you are interested in leading the adoption or tuning agility, this session will help prepare you for your first coaching gig.

### **Architecture and Agility Are Not Enemies by David Hussman**

Being agile does not mean living life one iteration at a time. Agile projects without a long view can run into the common design problems of the past. Planning iteration by iteration is often foolish and feeds the myth that agile projects do not think beyond a few weeks. Successful agile projects plan within iterations and across iterations. The later planning is called release planning and it is the forum where agility first engages architecture and other cross cutting concerns.

### **Open Source Debugging Tools for Java by Matthew McCullough**

This session will survey a wide range of tools across the Java space. We'll look at utilities such as VisualVM, jstatd, jps, jhat, jmap, Eclipse Memory Analyzer, jtracert, btrace and more. Open Source is not just a suite of libraries you consume within your application, but now reaches into the space of tools to help you troubleshoot and improve your applications. The price of these tools eliminates barriers to their use and their open source nature allows you to mix and match them into compositions that work well for your application's unique debugging needs.

### **Open Source Debugging Tools for Web Apps by Matthew McCullough**

This session will survey a wide range of tools across the Web application debugging space, covering the REST, HTML, SOAP, CSS, TCP, Filesystem and JavaScript facets of an app. We'll look at utilities such as tcpdump, curl, Wireshark, JMeter, Firebug, JASH, Poster, SoapUI, Firediff, Isof, fs\_usage, iwatch and more. Open Source is not just a suite of libraries you consume within your application, but now reaches into the space of tools to help you troubleshoot and improve your applications. The price of these tools eliminates barriers to their use and their open source nature allows you to mix and match them into compositions that work well for your application's unique debugging needs.

### **Git Going with Distributed Version Control by Matthew McCullough**

Many development shops have made the leap from RCS, Perforce, ClearCase, PVCS, CVS, BitKeeper or SourceSafe to the modern Subversion (SVN) version control system. But why not take the next massive stride in productivity and get on board with Git, a distributed version control system (DVCS). Jump ahead of the masses staying on Subversion, and increase your team's productivity, debugging effectiveness, flexibility in cutting releases, and repository redundancy at \$0 cost. Understand how distributed version control systems are game-changers and pick up the lingo that will become standard in the next few years.

**Prerequisite:** *Basic understanding of Subversion or similar version control system*

### **Mastering Maven 2.0 by Matthew McCullough**

Maven has been on the Java build tools scene for quite a number of years, but the adoption rate in enterprises is now going through the roof. Maven can seem daunting, but this presentation will equip existing Maven users with more efficient techniques and tools to overcome the biggest perceived Maven hurdles and build issues with ease. We'll examine tools to help you find artifacts in central repositories, manage your corporation's internal Maven artifacts with a proxy tool such as Nexus, view and override dependency graphs, dependency management and multi-module best practices, create OS specific profiles, and leverage the latest Maven plugins for the top Java IDEs. **Prerequisite:** *Basic Maven knowledge*

### **Cloud Computing Boot Camp on the Google App Engine by Matthew McCullough**

Cloud this, cloud that. It's all we are hearing about these days. And whether buzz-worthy or not, you need to get in-the-know so that you can talk effectively about how this could fit into the application strategy on your next project. Leverage 100s of hours of research distilled into a 90 minute presentation. Get bootstrapped

with what cloud computing is and isn't, who the players are in this space, what unique features each offers, and then how Google is completely changing the game.

### **The Busy Java Developer's Guide to Java Platform Security by Ted Neward**

Permissions, policy, SecurityExceptions, oh my! The Java platform is a rich and powerful platform, complete with a rich and powerful security mechanism, but sometimes understanding it and how it works can be daunting and intimidating, and leave developers with the basic impression that it's mysterious and dark and incomprehensible. Nothing could be further from the truth, and in this presentation, we'll take a pragmatic, code-first look at the Java security platform, including Permissions, the SecurityManager and its successor, AccessController, the Policy class and policy file syntax, JAAS, and more.

### **The Busy Java Developer's Guide to Advanced Platform Security by Ted Neward**

So you know the platform security model, and now you want to use it in new and interesting ways, like creating a custom Policy implementation, a custom Permission, or create a custom security context in which code will execute. Perhaps you even wish to make certain objects accessible only to those with the right permissions, or cryptographic key. Nothing could be easier, despite Java security's reputation as a dark and arcane place. **Prerequisite:** *The Busy Java Developer's Guide to Platform Security*

### **The Busy Java Developer's Guide to ClassLoaders by Ted Neward**

If you've ever gotten a ClassCastException and just knew the runtime was wrong about it, or found yourself copying .jar files all over your production server just to get your code to run, then you probably find the Java ClassLoader mechanism to be deep, dark, mysterious, and incomprehensible. Take a deep breath, and relax--ClassLoaders aren't as bad as they seem at first, once you understand a few basic rules regarding their operation, and have a bit more tools in your belt to diagnose ClassLoader problems. And once you've got that, and hear about ClassLoaders' ability to run multiple versions of the same code at the same time, and to provide isolation barriers inside your application, or even compile code on the fly from source form, you might just find that you like ClassLoaders after all... maybe.

### **The Busy Developer's Guide to Iconoclasm by Ted Neward**

History is littered with the stories of iconoclasts--people who truly stood out as pioneers, lateral thinkers, and in some cases, outright heroes--and their successes and failures. From the baseball management vision of Branch Hickey to the glassblowing vision of Dale Chihuly to the engineering design vision of Steve Jobs, iconoclasts have changed our world in subtle and profound ways, sometimes loudly, sometimes quietly.

### **The Busy Java Developer's Guide to Hacking with the JDK by Ted Neward**

Ever since its 1.1 release, the Java Virtual Machine steadily becomes a more and more "hackable" (configurable, pluggable, customizable, choose your own adjective here) platform for Java developers, yet few, if any, Java developers take advantage of it. Time to take the kid gloves off, crack open the platform, and see what's there. Time to play.

### **The Busy Java Developer's Guide to Collections by Ted Neward**

For so many Java developers, the java.util.\* package consists of List, ArrayList, and maybe Map and HashMap. But the Collections classes are so much more powerful than many of us are led to believe, and all it requires is a small amount of digging and some simple exploration to begin to "get" the real power of the Collection classes.

### **The Busy Java Developer's Guide to Functional Java by Ted Neward**

Much noise has been made in recent years about functional languages, like Scala or Haskell, and their benefits relative to object-oriented languages, most notably Java. Unfortunately, as wonderful as many of those benefits are, the fact remains that most Java developers will either not want or not be able to adopt those languages for writing day-to-day code. Which leaves us with a basic question: if I can't use these functional languages to write production code, is there any advantage to learning about them? The short answer is yes, for the fundamental premise--"I can't use functional code on my Java project"--is flawed. Java developers can, in fact, make use of functional ideas, and what's better, they don't even have to reinvent them for Java--thanks to the FunctionalJava library, many of the core primitives--interfaces that serve as base types for creating function values, for example--already exist, ready to be used.

### **The Busy Java Developer's Guide to Advanced Collections by Ted Neward**

Once you've learned the core Collections classes, you're done, right? You know everything there is to know about Collections, and you can "check that off" your list of Java packages you have to learn and know, right?

**Prerequisite:** *Busy Java Developer's Guide to Collections*

### **Real-world JEE performance tuning: Tips n' Tricks by Pratik Patel**

Performance tuning any application is a black art that can consume much time. Fortunately, Java has many tools that can aid in this effort. There also are a number of basic tips that can help to analyze and fix performance problems. The Java memory model is usually something that you don't need to tune, but for high performance applications it is necessary to tweak. While there are a number of advanced things that can be done to performance tune an application, we'll discover that the simple, basic things are all that are usually needed to make your apps fly.

### **Enterprise JPA & Spring 3.0 - Tips and Tricks for JEE5 Persistence by Pratik Patel**

As with many technologies, the basics are easy. The hard part comes when the developer needs to do sophisticated integration, development, and testing as part of an enterprise application. A large enterprise application requires the developer to think of issues that affect the development, scalability and robustness of the application. This presentation will cover the advanced topics described below with a focus on the new persistence features in Spring 3.0 and JPA 2.0.

### **JUnit4: Take your testing to the next level by Pratik Patel**

JUnit 4 is the latest release for the venerable unit testing framework used by all Java developers. JUnit 4 introduces many enhancements and features over JUnit 3. This session focuses on using these new features. Code examples will drive this presentation to help the attendee understand the best way to start using the new features. We'll also cover core unit testing principles to refresh everyone's understanding of unit testing best practices.

### **Groovy and Grails in the Enterprise by Pratik Patel**

Dynamic languages running on the Java Virtual Machine are starting to gain traction for software development, specifically for large enterprise projects. This session explores obstacles to introducing dynamic languages into the enterprise, example applications that can ease the way, and issues surrounding integrating a dynamic language to Java projects. Using several code examples that demonstrate the power of using a dynamic language like Jruby or Groovy, attendees will gain insight into how dynamic languages are making in-roads to the enterprise. This session focuses on non-GUI related usages whereas most people think of dynamic languages for Web development. The target audience for this session is enterprise developers and enterprise architects.

### **Virtualization for development by Pratik Patel**

We've all heard about virtualization for deploying applications. How about leveraging virtualization for development? In this session, we'll look at some time saving tips and build a virtual VM for development and testing.

### **Common AntiPatterns and How To Avoid Them by Mark Richards**

In the book "97 Things Every Software Architect Should Know" (O'Reilly, 2009) I wrote about the importance of design patterns as a useful means of communication between architects and developers. Equally important to patterns is an understanding of AntiPatterns - things that we repeatably do that produce negative results. AntiPatterns are used by developers, architects, and managers every day and are one of the main factors that prevent progress and success. In this session we will look at some of the more common and significant development and architecture antipatterns. Through coding and design examples, you will see how these antipatterns emerge, how to recognize when the antipattern is being used, and most importantly, how to avoid them. By attending this session, you will be part of a movement to reduce the AntiPattern catalog from hundreds of entries to only a few. **Prerequisite:** *None*

### **On Being a Software Architect by Mark Richards**

One way to stop a conversation dead while at a party or gathering is to mention you are a software architect. Why? Because it takes about an hour (complete with Powerpoint slides) to explain what you do for a living. By then the person you are talking to is so bored they would rather sit in a corner licking nine-volt batteries. The problem is that no one inside or outside our industry really knows what a software architect is or what they do. In this highly interactive (and slightly humorous) session we will take a deep dive into the role a software architect plays in the IT industry. We will explore the characteristics an architect needs to have,

and the elements that make a good architect and a bad architect. Through amusing antidotes and real-world examples, we will see how to become an effective software architect and help shape the industry in terms of the role and title of software architect. **Prerequisite:** *None*

### **Transaction Pitfalls and Strategies by Mark Richards**

In previous years I have given sessions related to my book "Java Transaction Design Strategies", where I have reviewed the basics of programmatic and declarative transactions and outlined the basic patterns described in the book. In this new session for 2009 I will focus on some of the pitfalls encountered while dealing with transactions and then how to develop an effective transaction strategy. I will start this session by describing and illustrating some of the common pitfalls I continue to see in both Spring and EJB. I will then describe four common transaction strategies you can use and implement, including a transaction strategy for high-speed transactions, a transaction strategy for client orchestration, a transaction strategy for use with API's, and finally a strategy for highly concurrent environments. Note: This session assumes you know a little bit about transactions and have been using them in either Spring or EJB. It is not intended to be an introductory session on how transactions work. You can obtain a free PDF download of my transaction book at <http://www.infoq.com/minibooks/JTDS> to quickly come up to speed with transactions. **Prerequisite:** *Java, Spring or EJB; some knowledge of transactions and JTA.*

### **Introduction to JMS by Mark Richards**

There's no doubt about it - messaging is quickly becoming a standard part of most application architectures, particularly as more and more companies struggle to find ways to integrate heterogeneous environments due to mergers, acquisitions, or to streamline existing application portfolios. The Java Message Service (JMS) API allows Java applications to implement messaging using a standard API, therefore removing the dependency of any particular messaging provider. In this introductory session we will take a look at the basics of messaging and the JMS API. I will start by discussing the different messaging models, the structure of a basic JMS message, and the JMS API interfaces and how they interrelate. Then through interactive coding I will show the basics of sending and receiving messages using the point-to-point messaging model and how to do request/reply processing. NOTE: this session is meant to be an introduction to messaging and JMS - no prior JMS or messaging experience is needed for this session. **Prerequisite:** *None*

### **Advanced Topics in JMS by Mark Richards**

This session covers some of the more advanced features of JMS messaging, and is intended for those who are familiar with JMS and messaging in general. Some of the topics I will be covering in this session include message grouping (where I will demonstrate sending a large JPG image using messaging), transacted sessions, client-based acknowledgement, and some various messaging design considerations and things to watch out for from a design and coding perspective. I will be doing live coding demonstrations to illustrate the techniques described in this session. Although this session is entirely JMS provider agnostic, I will be using ActiveMQ, a popular open source JMS provider, during the live coding demonstrations. **Prerequisite:** *Some knowledge of messaging and JMS would be helpful*

### **Spring and JMS: Message-Driven POJOs by Mark Richards**

The Java Message Service (JMS) provides a standard messaging API that allows you to send and receive messages using a variety of messaging providers (including Java EE application servers). The Spring Framework takes this abstraction one step further by providing a robust JMS messaging framework that greatly simplifies message processing. In this session we will see how to use the JMS Messaging Framework provided in Spring 2.5. I will start by describing Spring's overall messaging architecture and how to configure the various beans needed for messaging. Then, through interactive coding I will discuss and demonstrate Spring's JMS Template, which is used for sending messages and receiving messages synchronously. I will then discuss and demonstrate Message Driven POJOs, which are Spring's answer for asynchronous message listeners. After attending this session you will have all the necessary knowledge and code examples to use JMS in your Spring applications. **Prerequisite:** *Knowledge of JMS and Spring*

### **REST : Web Architecture for Rich Clients by Brian Sletten**

The failure of a Service-Only Architecture (SOA) is that it fails to highlight the data that flows through it. We must embrace a software architecture that puts information first. Who wants what? How do they want to use it? This blended vision that handles data, documents, services and concepts There is tremendous interest in REpresentational State Transfer (REST) as an architectural style for building scalable, flexible, information-driven architectures in the Enterprise. The success of the Web has caught our attention in the face of increased complexity and many failures with more traditional Web Services technologies. The

problem is that it is difficult to sell a way to do things. Managers do not want to feel like they are innovating in the middleware space. They want to understand why they should deviate from the blue prints laid down by the industry leaders. They want to understand when they should use REST, when they should use SOAP and when they might fallback to regular old Java-based messaging. They want to make business-based technology decisions that lay a path to forward progress rather than paying for technological flux.

### **RDFa : Weaving Richness and Meaning in the Web by Brian Sletten**

The human web is reasonably well in hand by now. We are getting pretty good at building systems that people find valuable and entertaining. We have not spent as much time concerned about our software friends. There is a ton a rich content available on the web that is too difficult to extract in automated ways using just XHTML, the meta tag and microformats. This talk will introduce you to some emerging technologies from the Semantic Web camp to enrich your web pages with useful information for both automated extraction and improved browsing experiences.

### **SPARQL: Querying the Data Web by Brian Sletten**

The human-friendly Web is about nicely-formatted, accessible content for users to browse. There is an emerging Data Web that relies on technologies from the Semantic Web stack to link increasingly rich connections between various data sources. SPARQL and RDF are the main tools for expressing and using this connectivity. This talk will introduce you to one of the practical and accessible aspects of employing these ideas on the Web and in the Enterprise. *Prerequisite: The Semantic Web: The Future, Now and Rich Web Pages : Publishing Semantic Content with GRDDL and RDFa would both be helpful but are not required*

### **Semantic SOA : Meaningful Service Strategies by Brian Sletten**

The goal for web services was always to reduce our burden by increasing the potential for reuse of business functionality. Somehow, we got lost along the way in a morass of confusing, unfulfilling and downright broken technologies. While we are interested in pursuing REST-based systems for managing information, we need some strategies for tying it all together sensibly. If we abandon WSDL, SOAP and UDDI, what do we replace them with? This talk will walk you through combining resource-oriented strategies with technologies from the Semantic Web to describe, find, and bind to services in dynamic, flexible and extensible ways.

*Prerequisite: The Semantic Web: The Future Now, Give it a REST and SPARQL : Querying the Data Web would all be helpful talks to have attended*

### **Groovy for Java Programmers by Venkat Subramaniam**

This fast paced presentation is intended for experienced Java programmers. You will start by learning what Groovy is.

### **Cleaning up Code Smell by Venkat Subramaniam**

Projects often start out simple, but soon become complex and turn into a lose cannon. Organizations are struggling to maintain and evolve software. Poor code quality is a significant part of that problem. Improving the quality of code is critical to success of enterprise projects.

### **Testing with dependencies by Venkat Subramaniam**

Testing is a key ingredient to the success of a project. However, testing becomes awfully hard when your application deals with dependencies and that is often the reality.

### **Programming Scala by Venkat Subramaniam**

Scala is a static fully object-oriented, functional language on the JVM. While taking advantage of the functional aspects, you can continue to make full use of the powerful JVM and Java libraries.

### **Tackling Concurrency on the JVM by Venkat Subramaniam**

In this presentation we will take a quick walk though the issues with concurrency and how the solutions provided in Scala and Clojure help address those.

### **Building External DSLs by Venkat Subramaniam**

Domain Specific Languages (DSLs) are languages targeted at a particular problem and domain. They have context and are fluent. They help users of applications at various levels to easily communicate with your

application. Developing DSLs, however, are not easy. You could easily get dragged into using parsers and tools with steep learning curve.

### **Effective Java by Venkat Subramaniam**

Java is a well established language, that has been around for more than a decade. Yet, programming on it has its challenges. There are concepts and features that are tricky. When you run into those, the compiler is not there to help you.