

Northern Virginia Software Symposium

Sheraton Reston

April 24 - 26, 2009

<http://www.nofluffjuststuff.com/conference/reston/2009/04/index.html>

Fri, Apr. 24, 2009

	Meeting Room 9 & 10	Meeting Room 4	Meeting Room 8	Meeting Room 7	Meeting Room 6	Meeting Room 5
12:00 - 1:00 PM	REGISTRATION					
1:00 - 1:15 PM	WELCOME					
1:15 - 2:45 PM	Emergent Design & Evolutionary Architecture Neal Ford	Introduction to JMS Mark Richards	Restoring Agility: Getting Your Team Back on Track Jared Richardson	The Amazing Groovy Weight-loss Plan Scott Davis	The Busy Java Developer's Guide to Java7 Ted Neward	JSF 2.0: An Introduction David Geary
2:45 - 3:15 PM	BREAK					
3:15 - 4:45 PM	Real-world Refactoring Neal Ford	Advanced Topics in JMS Mark Richards	Software Team Tuneup Jared Richardson	Groovy XML Ninja Skills Scott Davis	The Busy Java Developer's Guide to Java Platform Security Ted Neward	JSF 2.0: Advanced Topics David Geary
4:45 - 5:00 PM	BREAK					
5:00 - 6:30 PM	Test Driven Design Neal Ford	Transaction Pitfalls and Strategies Mark Richards	Agile Anti-Patterns Jared Richardson	DSLs in Groovy: Say What You Mean, Mean What You Say Scott Davis	The Busy Java Developer's Guide to Advanced Platform Security Ted Neward	Flex for Java Developers David Geary
6:30 - 7:15 PM	DINNER					
7:15 - 8:00 PM	Keynote: by Neal Ford					

Sat, Apr. 25, 2009

	Meeting Room 9 & 10	Meeting Room 4	Meeting Room 8	Meeting Room 7	Meeting Room 6	Meeting Room 5
8:00 - 9:00 AM	BREAKFAST					
9:00 - 10:30 AM	Spring 3.0 Overview Scott Leberknight	Agile, Relevance Style Stuart Halloway	Dim Sum Grails: A Sampler of Practical Non Database-Driven Grails Applications Scott Davis	Regular Expressions in Java Neal Ford	Techniques 2009 Jared Richardson	On Being a Software Architect Mark Richards
10:30 - 11:00 AM	BREAK					
11:00 - 12:30 PM	The Busy Java Developer's Guide to Collections Ted Neward	Taking Agile From Tactics to Strategy Stuart Halloway	Web 2.0 Checklist: Deconstructing Modern Websites Scott Davis	Construction Techniques for Domain Specific Languages Neal Ford	Real World Hibernate Tips (Reloaded) Scott Leberknight	Common AntiPatterns and How To Avoid Them Mark Richards
12:30 - 1:30 PM	LUNCH					
1:30 - 3:00 PM	Java.next #1: Common Ground Stuart Halloway	REST : Information-Driven Architectures for the 21st Century Brian Sletten	Lizard Brain Web Design Scott Davis	Visualizations for Code Metrics Neal Ford	The Reality of Continuous Availability Mark Richards	GWT: An Introduction David Geary
3:00 - 3:15 PM	BREAK					
3:15 - 4:45 PM	Architect for Scale Michael Nygard	RESTlet for the Weary Brian Sletten	Programming Clojure Stuart Halloway	Hands-on Agile Development Neal Ford	The Busy Java Developer's Guide to Hacking with the JDK Ted Neward	GWT: Advanced Topics David Geary
4:45 - 5:30 PM	BIRDS OF A FEATHER SESSION					

Sun, Apr. 26, 2009

	Meeting Room 9 & 10	Meeting Room 4	Meeting Room 8	Meeting Room 7	Meeting Room 6	Meeting Room 5
8:00 - 9:00 AM	BREAKFAST					
9:00 - 10:30 AM	Effective Concurrent Java Brian Goetz	Agile is Dead, Long Live Agility David Hussman	The 90-Minute Startup Michael Nygard	Meta-programming JRuby for Fun & Profit Neal Ford	Groovy Closures - The way to cleaner code Mark Johnson	Project Integrity, Part 1: tools and techniques for managing source code quality David Bock
10:30 - 11:00 AM	MORNING BREAK					
11:00 - 12:30 PM	Are All Web Applications Broken? Brian Goetz	Producing Software Groove David Hussman	SPARQL: Querying the Data Web Brian Sletten	The Productive Programmer: Mechanics Neal Ford	Clouds, Grids, and Fog Michael Nygard	Capistrano: Application Deployment and More David Bock
12:30 - 1:15 PM	LUNCH					
1:15 - 2:15 PM	EXPERT PANEL DISCUSSION					
2:15 - 3:45 PM	Garbage-collector-friendly programming Brian Goetz	What Is Lean And Why Do You Care? David Hussman	Semantic SOA : Meaningful Service Strategies Brian Sletten	Google Your Domain Objects With Hibernate Search Scott Leberknight	Creating a real world application in Grails - A Case Study Mark Johnson	Intermediate Maven David Bock
3:45 - 4:00 PM	BREAK					
4:00 - 5:30 PM	The Java Memory Model Brian Goetz	Architecture and Agility Are Not Mutually Exclusive David Hussman	XMPP For the People : Smack and OpenFire Brian Sletten	Polyglot Persistence Scott Leberknight	The Software Manager's Dashboard: Getting the information you really need Mark Johnson	Surviving Middle Management David Bock

Northern Virginia Software Symposium

Sheraton Reston
April 24 - 26, 2009

<http://www.nofluffjuststuff.com/conference/reston/2009/04/index.html>

Project Integrity, Part 1: tools and techniques for managing source code quality by David Bock

How many times have you started a new project only to find that several months into it, you have a big ball of code you have to plod through to try to get anything done? Have you ever been the 'new guy' on a project where it seems like the code grew more like weeds and brambles than a well-tended garden? With a few good tools to help analyze the code, we can keep our project from turning into that big ball of mud, and we can salvage a project that is already headed down that path.

Capistrano: Application Deployment and More by David Bock

Capistrano (formerly Switchtower) is a tool originally written to help automate application deployment for Ruby on Rails. It does this well, but it has grown up into a tool capable of much, much more. It can be used for deploying Java applications, updating server configurations across an enterprise, administering networks, backing up files, and all sorts of other activities. Any activity you might do from the command line, you can now do simultaneously across large numbers of machines, with all machines succeeding (or rolling back in case of failure) together.

Intermediate Maven by David Bock

Maven is a build tool that does a lot, demos well, and leaves the build maintainers managing what seems like unbridled complexity. It doesn't have to be that way - Maven is driven by some strong 'build process methodology', and that complexity can become manageable by wrapping your head around it. Furthermore, you can migrate to Maven 'piecemeal', by mapping your existing ant build to the Maven Lifecycle and calling your existing Ant tasks - you can decide to sip the Maven kool-aid. Ideally, a build tool should be so simple and approachable that it fades into the project background and allows anyone to maintain it. Unfortunately, Maven's power comes at the expense of this ideal - Maven's philosophy is more like "the build process is so important that the people maintaining it should be steeped in the ways of Maven". This talk will give you the exposure you need without elevating The Maven Way to a religion.

Surviving Middle Management by David Bock

Most good developers eventually have the opportunity to be managers. Whether they call you the "project manager", "Technical Lead", "Lead Developer", or some other classic middle-management title, you become the 'goto' guy between management and developers. You're the guy who is expected to keep the project in-line, track a schedule, and occasionally answer the question "How's it going?", and perhaps still contribute at a technical level. So how do you do that?

The Amazing Groovy Weight-loss Plan by Scott Davis

"The central enemy of reliability is complexity." (Dr. Daniel Geer) Java is a powerful programming language. A smart developer can do nearly anything with Java. So the next question is, "How quickly can it be done? How many lines of code does it take to do common tasks?" Groovy greases the wheels of Java by decreasing the complexity of the language while preserving the raw power. At first glance, you might think that this talk is simply about how Groovy drastically reduces the lines of code you need to write. What this talk is really about is bringing simplicity, clarity, readability, and yes, beauty to your source code.

Groovy XML Ninja Skills by Scott Davis

"XML is like violence: if it doesn't solve your problem, you aren't using enough of it." (Anonymous) XML is everywhere. Whether you are dealing with local configuration files (web.xml, struts-config.xml) or remote web services (SOAP, REST, RSS, Atom), the modern software developer needs to be able to request, slice, and dice XML with ease. That requires a set of razor-sharp tools that reduce the inherent complexity of the problem, not multiply it. Once you see XML tremble in fear at the awesome power of Groovy, you'll wonder what you ever did without it.

DSLs in Groovy: Say What You Mean, Mean What You Say by Scott Davis

"Simplicity is the ultimate sophistication." (Leonardo da Vinci) The history of computer programming has been bridging the gap between what the user says ("We need to add sales tax to each item in the order") and what the programming language requires you to say ("for Iterator i = orderList.iterator();"). Building

Domain Specific Languages (DSLs) allow you to express the solution in the language of the domain user instead of the language of the programmer.

Dim Sum Grails: A Sampler of Practical Non Database-Driven Grails Applications by Scott Davis

"The proof of the pudding is in the eating. By a small sample we may judge of the whole piece."

(Miguel de Cervantes Saavedra) Most Grails tutorials demonstrate how easy it is to build simple CRUD (Create/Retrieve/Update/Delete) applications. While skinning a database with a web front-end is undeniably one beneficial aspect of Grails, it isn't the only thing Grails is good for. As you'll see here, Grails can be used to build a wide variety of web applications. You won't see a single HTML table with "edit" and "delete" links, I promise.

Web 2.0 Checklist: Deconstructing Modern Websites by Scott Davis

"The challenge of modernity is to live without illusions and without becoming disillusioned." (Antonio Gramsci) There are plenty of sarcastic "Web 2.0" checklists out there -- be perpetually in BETA, when in doubt add rounded corners, etc. While we can all laugh at the superficial aspects of the Web 2.0 revolution, there are plenty of serious aspects to it as well. Is your website mash-up friendly or hostile? Do you tell your visitors when things change (via RSS or Atom syndication), or do you expect them to check in daily for updates? Is your website a silo or a part of a larger ecosystem?

Lizard Brain Web Design by Scott Davis

"There's an old story about the person who wished his computer were as easy to use as his telephone. That wish has come true, since I no longer know how to use my telephone." (Bjarne Stroustrup) The "lizard brain" is the oldest part of the human brain -- the part responsible for autonomic functions like breathing, heart rate, and navigating websites. OK, maybe not that last part, but your website should be easy to use. Stupid easy. Lizard brain easy. Any time your user spends figuring out how to do something -- even for a split second -- is wasted time due to poor design. Inspired by Steve Krug's book "Don't Make Me Think", this talk answers the question, "Why is that website so hard to use?"

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 cruddy 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.

Test Driven Design by Neal Ford

Most developers think that "TDD" stands for Test-driven Development. But it really should stand for "Test-driven Design". Rigorously using TDD makes your code much better in multiple ways.

Keynote: On the Lam from the Furniture Police by Neal Ford

When you were hired by your current employer, you may think it's because of your winning personality, your dazzling smile, or your encyclopedic knowledge of [insert technology here]. But it's not. You were hired for your ability to sit and concentrate for long periods of time to solve problems, then placed in an environment where it's utterly impossible to do that! Who decides that, despite overwhelming evidence that it's bad for productivity and people hate it, that you must sit in a cubicle? The furniture police! This keynote describes the frustrations of modern knowledge workers in their quest to actually get some work done, and solutions for how to gird yourself against all those distractions. I talk about environments, coding, acceleration,

automation, and avoiding repetition as ways to defeat the mid-guided attempts to sap your ability to produce good work. And I give you ways to go on the lam from the furniture police and ammunition to fight back!

Regular Expressions in Java by Neal Ford

Regular expressions should be an integral part of every developer's toolbox, but most don't realize what an important topic it is. Regular expressions have existed for decades, but many developers don't understand how to take full advantage of this powerful mechanism, either through command line tools and editors or in their development.

Construction Techniques for Domain Specific Languages by Neal Ford

This talk covers language techniques in Java, Groovy, and Ruby on how and why to create DSLs, and also covers the very important topic of implicit context, and how language constructs can allow you to write less verbose and more expressive code.

Visualizations for Code Metrics by Neal Ford

Judicious use of metrics improves the quality of your code. But interpreting metrics presents a challenge. You have a list of numbers for a project - what does it mean? And what does it tell me about the health of the project overall? This sessions shows how to produce visualizations for software metrics, making them easier to understand and more valuable. It covers metrics at the individual method level all the way up to the overall architecture of the application. This isn't just a talk about how some tools produce visualizations: this session shows you how to generate your own visualizations, allowing you to customize it to the level in information density that shows real value on your project. I show how to produce projected graphs from dependencies, heat-maps for cyclomatic complexity and code coverage, using XSLT to extract visual information from XML configuration documents, and others. Metrics can't help you if you can't understand them. By creating visualizations, it helps leverage metrics to make your code better.

Hands-on Agile Development by Neal Ford

BRING YOUR LAPTOP WITH YOU, BUT A LAPTOP ISN'T REQUIRED! Reading and hearing about agile practices is one thing, but actually doing it is completely different. This session puts you to work in an agile fashion, applying agile developer practices.

Meta-programming JRuby for Fun & Profit by Neal Ford

Ruby is the revenge of the Smalltalkers. Not since Smalltalk has a language had such powerful meta-programming facilities. While this may seem like a minor feature, it turns out that surgical meta-programming allows solutions to problems that are clearer, more concise, more maintainable, and take orders of magnitudes fewer lines of 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!

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 advanced aspects of JSF 2.0. **Prerequisite:** *Familiarity with JSF, or other component-based frameworks. Familiarity with Ajax. This session builds on demos shown in 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: An Introduction by David Geary

An introduction to Google Web Toolkit. **Prerequisite:** *Familiarity with a component-based framework, preferably a desktop application framework*

GWT: Advanced Topics by David Geary

Learn to do really cool stuff with GWT. **Prerequisite:** *The GWT: Introduction session 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.

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*

Garbage-collector-friendly programming by Brian Goetz

To many developers, garbage collection is black magic. Accordingly, there are a lot of conflicting advice about what is good or bad for the garbage collector. In this talk, I look at how garbage collection is implemented in the HotSpot VM, and techniques for writing programs that exhibit good garbage collection behavior. Surprisingly, many of these techniques coincide with writing good, clean code.

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.

Agile, Relevance Style by Stuart Halloway

The Agile Manifesto, like any good scripture, admits of many interpretations. There is no one "right path." What works for us may not work for you. At Relevance we have tried many paths, and learned many lessons. Join us to see dozens of ideas that have worked for us, plus some that haven't.

Taking Agile From Tactics to Strategy by Stuart Halloway

Teams adopting agile should begin at a tactical level, but they shouldn't end there. The Agile Manifesto operates at many different levels. Learn to apply the principles of agile at a strategic level. Otherwise you can have a great agile ground game and still lose.

Java.next #1: Common Ground by Stuart Halloway

In this talk, we will explore and compare four of the most interesting new JVM languages: Clojure, Groovy, JRuby, and Scala. Each of these languages aims to greatly simplify writing code for the JVM, and all of them succeed in this mission. However, these languages have very different design goals. We will explore these differences, and help you decide when and where these languages might fit into your development toolkit. For more information see <http://blog.thinkrelevance.com/2008/8/4/java-next-common-ground>.

Programming Clojure by Stuart Halloway

Find out why Clojure is Java.next: * Clojure provides clean, fast access to all Java libraries. * Clojure provides all the low-ceremony goodness you know and love from dynamic languages such as Ruby and Python. * Clojure includes Lisp's signature feature: Treating code as data through macros. * Clojure's emphasis on immutability and support for software transactional memory make it a viable option for taking advantage of massively parallel hardware.

Agile is Dead, Long Live Agility by David Hussman

If you are truly working to get value from agile methods, and you have been doing it for some time, you probably have some questions which go beyond "my first agile project". If you are looking for a place to get answers or hear where and how others are struggling, come to this session ready to ask your tough

questions. I have coached many communities (of all shapes and sizes) adopt, adapt, and evolve agility beyond the first project or the first few months, and I am sure there will be no shortage of examples and experiences for you questions. Also, I am sure you will learn from others in the audience as well.

Producing Software Groove by David Hussman

Where software production is successful, there are people drawing out common strengths and raising awareness around unhealthy habits. These software producers, often called "coaches", possess named and unnamed skills of all types. You will learn ideas for producing (coaching) the adoption and the adaption of pragmatic agility. You will learn common and uncommon practices as we work with (and get to know) several examples based on actual communities with varying degrees of experience using agile methods.

What Is Lean And Why Do 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.

Architecture and Agility Are Not Mutually Exclusive 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.

Groovy Closures - The way to cleaner code by Mark Johnson

The factory patterns and callbacks have been around for a long time as a technique to provide flavor specific code variations. But they are awkward and hard to update. Enter Groovy closures. Imagine having the ability to inject different coding flavors using code closures. If you need a different flavor, then just pass a different code block. Now imagine that all of this works on the JVM!

Creating a real world application in Grails - A Case Study by Mark Johnson

You have probably heard about Grails, a Rails based development framework which claims to make software development a breeze. You have heard the hype and are now wondering about the reality, will it really work in your environment? Has it really lived up to the hype? This presentation reviews the results using Grails to develop the New England Java Users Group event management web site to answer these questions.

The Software Manager's Dashboard: Getting the information you really need by Mark Johnson

When we start a project, our management hands us a copy of MS Project and using this tool we are expected to accurately track the project to completion. What often ends up happening is many of the project tasks are listed as 90% complete and you don't have a clear understanding of the blocking reasons. This presentation will explore various vendor independent time efficient dashboard options you can pursue to properly track your project.

Spring 3.0 Overview by Scott Leberknight

The Spring framework has simplified Java enterprise and web development since 2003, and has been a major innovator in improving and simplifying Java server-side programming since its inception. This session will look at the new features in Spring 3.0 as well as what's being removed from the Spring core.

Real World Hibernate Tips (Reloaded) by Scott Leberknight

Hibernate is a very powerful object/relational mapping framework. This session contains a new set of Hibernate tips, tricks, and pitfalls.

Google Your Domain Objects With Hibernate Search by Scott Leberknight

Hibernate is one of the pre-eminent object/relational mapping technologies, but the Hibernate Search project adds full-text search capabilities to an already extremely capable tool to allow you to Google your domain objects.

Polyglot Persistence by Scott Leberknight

Polyglot persistence is all about considering your persistence requirements and selecting a persistence mechanism that best meets those requirements, as opposed to selecting an RDBMS as the default choice. In this session we'll look at some of the persistence alternatives that are available like Amazon SimpleDB, CouchDB, Google Bigtable, and more.

The Busy Java Developer's Guide to Java7 by Ted Neward

Even though the Java 7 JSR has yet to be formed, some interesting things are beginning to emerge from Sun about what Java7 may include when its formal release contents are finally made public.

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 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 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.

Architect for Scale by Michael Nygard

Is your system small, medium, large, or super-size? Is traffic on its way up? Architecture patterns and structures that work at one scale seldom work across all of them. A communication style that's appropriate for small websites will probably fail badly if you apply it to world-wide networks of computers. Likewise, structures that work for large-scale systems are probably too complex and expensive to be worth it for small sites.

The 90-Minute Startup by Michael Nygard

Cloud computing is taking the world by storm. Amazon's Web Services, EC2, and S3 provide completely virtual infrastructure, letting startup and existing companies create sites and web applications faster than ever before. In this session, Michael will use cloud computing to create and deploy a fully-functional web site. You will learn how to create and run your own virtual infrastructure in the clouds.

Clouds, Grids, and Fog by Michael Nygard

Servers, storage, networking, backups... they're all vanishing into the "clouds". Cloud Computing is the emerging architecture for massive, scalable infrastructure that your company doesn't have to own or operate. From the "zero servers" web startup to the corporate IT department battling server-sprawl, cloud computing

has many manifestations. This session will differentiate among the various types of cloud computing and describe applicable use cases.

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 and batching (where I will demonstrate sending a very large JPG image using messaging), transacted sessions, client-based acknowledgement, message correlation alternative, 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*

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.*

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*

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

The Reality of Continuous Availability by Mark Richards

Ever wonder how to accurately calculate high availability and continuous availability? Ever wonder the real difference between clustered and active/active topologies? Ever wonder why the extraordinary cost and effort to put a CA environment in place rarely yields the expected results? Ever wonder how to make CA work without having to sit through vendor presentations or demos? Ever wonder why businesses not needing the "6 nines" of availability commonly found with CA environments are still pursuing CA? Ever wonder if there's more to CA than numbers, calculations, and topologies? Ever wonder what the future holds for continuous availability computing? While some industries such as Telecommunications have solved the continuous availability issue, other industries such as banking, insurance, and financial markets still find continuous availability a challenging and complex task. These industries typically have a complex and heterogeneous assortment of technologies, platforms, and architecture layers which make designing and implementing continuous availability particularly challenging. Introduce Service Oriented Architecture into the mix, and the issue becomes even more complex. Come to this vendor-agnostic session to see some of the answers to the mysteries surrounding the black art of continuous availability; all is not what it seems....

Restoring Agility: Getting Your Team Back on Track by Jared Richardson

An agile team is first and foremost "a team". When that gets lost in the rush to get a product out the door, the people suffer as well as the products. It's bad for the company, but even worse for the team members. We'll learn how to defuse some of the more common problems you'll run into on dysfunctional teams.

Software Team Tuneup by Jared Richardson

We're always under pressure to do more with less. More features with less developers. More product in less time. More work in fewer hours.

Agile Anti-Patterns by Jared Richardson

Agile is wildly popular in some circles and hated in others. How can the same ideas cause such different reactions? Sometimes it's the definition of "agile" and other times it's company culture, but there's usually a good reason when Agile ideas are thrown out on their collective ears.

Techniques 2009 by Jared Richardson

There are a number of great techniques you can use across technologies and projects. Come hear some of my favorite ways to move "beyond" and contribute a few of your own. We'll discuss topics ranging from glue languages to ditching your IDE to building your brain.

REST : Information-Driven Architectures for the 21st Century by Brian Sletten

There is a shift going on in the Enterprise. While still used and useful, the promises of the SOAP/WSDL/UDDI Service-Oriented Architecture (SOA) stack have failed to live up to their promise. A new vision of linked information is enveloping online and Enterprise users. The REST architectural style is squarely behind this thinking as a way of achieving low-cost, flexible integration, increased data security, greater scalability and long-term migration strategies. If you have dismissed REST as a toy or are unfamiliar with it, you owe it to yourself to see what is so interesting about this way of doing things.

RESTlet for the Weary by Brian Sletten

If you have started to take a look at REST as way of exposing web services or managing information spaces, you may be frustrated by the support offered by legacy containers. There is no direct support for REST concepts in the J2EE specs (yet). XML-based configurations are so 1990's. Come learn about Restlets, a little API that has caught the attention of many in the RESTafarian community. **Prerequisite:** REST (unless you are very comfortable with REST)

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*

XMPP For the People : Smack and OpenFire by Brian Sletten

Communication is a key piece of organizational success. We are constantly trying to find new ways to improve how we share ideas, concepts, issues and needs. E-mail has lost its luster. Phone calls are too disruptive. We want a combination of presence-aware and asynchronous. Instant messaging has started to fill this void but using public services are often inappropriate for sensitive conversations. This talk will focus on two open source offerings: the OpenFire XMPP (Jabber) server and the Smack client library. Come hear how you can adopt these technologies in your daily activities more fully.