

New York Software Symposium 2005

Sheraton Newark Airport - Newark, NJ

July 08 - 10, 2005

(session listing as of 7/5/2005)

The No Fluff Just Stuff Software Symposium 2005 tour is pleased to announce the New York Software Symposium coming to Newark on July 08 - 10, 2005. You will have the opportunity to attend the best technically focused Java / Open Source event anywhere. We make this claim based on the following:

- 1) **Excellent Speakers with unparalleled access**
- 2) **Limited Attendance = 250 Registrants Max**
- 3) **No Vendors, No Sales Pitches, No Marketecture**
- 4) **Unmatched Value - less than 1/3 of the cost of a national conference**
- 5) **Since 2002, we have delivered over fifty(50) conferences throughout North America**

The No Fluff Just Stuff Software Symposium Series caters to individual developers, development teams, project managers, architects and independent consultants. The New York Software Symposium will offer 6 concurrent sessions over three days with over 65 sessions to choose from. The following topics will be featured:

- 1) .Net
- 2) ServerSide Java
- 3) XML / Web Services
- 4) Architecture
- 5) Client Side Java
- 6) Core Java

Registration Fees

Attendees	Before 6/20/2005	After 6/20/2005
1-4	\$675	\$775
5-9	\$600	\$675
10-14	\$575	\$650
15-24	\$550	\$625
25+	\$525	\$600

The Registration Fee includes the following:

- 1) All Access Pass to the three day symposium
- 2) Handouts from all sessions attended w/binder
- 3) CD with all presentational content @ registration
- 4) Custom NFJS Laptop/Backpack
- 5) Opportunity to win an iPod everyday during the symposium raffle

Go to <http://www.nofluffjuststuff.com> for more details.

Questions/Comments: Contact Jay Zimmerman: jzimmerman@nofluffjuststuff.com or (303)469-0486.

New York Software Symposium 2005

Sheraton Newark Airport - Newark, NJ

July 08 - 10, 2005

(session listing as of 7/5/2005)

Advanced Swing: Architecture and Frameworks by Ben Galbraith

Are you spending more time plumbing your Swing applications than solving business problems? Has your Swing application grown out of control? This session is for you.

Creating Polished Swing Applications by Ben Galbraith

Too often, Swing applications are slow, ugly, and hard-to-maintain. It turns out that it doesn't have to be this way. Swing can be used to create highly-responsive, beautiful applications that are very maintainable. If this isn't consistent with your own experience, don't feel bad; it's not very obvious how to make Swing sing.

AJAX: Creating Next-Generation, Highly Dynamic, Off-line Capable Web Applications with HTML and JavaScript by Ben Galbraith

As recent high-profile web apps such as Google's GMail have shown, modern browsers are capable of natively rendering web apps with highly dynamic and compelling UIs - fetching server data without page refreshes, animating and manipulating page contents on-the-fly, even offline use. The line between web and "desktop" apps is blurring.

JmDNS : Easy Service Discovery for the 21st Century by Brian Sletten

Configuration files are so 90's. Software of the 21st Century should be able to find related services and components without users having to specify particular configurations at start up. The IETF's ZeroConf multicast DNS protocol was designed to solve exactly this problem. jmDNS is Java-based open source implementation of this capability that allows local-link applications to find and use automagically discovered capabilities. Apple's Rendezvous technology is another open-source ZeroConf implementation behind many of the exciting applications it is building for OS X these days. Come learn how you can interact with these or your own service discovery-savvy applications without even having to learn how to spell UDDI. Bring your wireless notebooks to participate in a service-oriented environment (please have a working Java environment as we won't have time to debug installation issues).

Applied Object-Oriented Metrics by Brian Sletten

Object-oriented code metrics are a little like Artificial Intelligence: those who did it twenty years ago roll their eyes at the thought and prophesy the same ultimate failure at applicability now. Those who grew up with Java are approaching the topic with new eyes and are finding useful ways of incorporating metrics into their projects. Come hear about tools and ways to measure properties of software, how they might be beneficial and where you are likely to go astray with this approach.

Applied Design Patterns by Brian Sletten

Just about every modern software developer has a copy of the Gang of Four "Design Patterns" book sitting on a shelf; many of them have actually read it. The dark secret of the patterns community is that there is often a large gulf between whiteboard simplicity and real-world complexity. Language choice plays a part in the design (and even importance) of patterns. The situation is made even more confusing by the fact that many of the core patterns have now been "voted off the island" for one reason or another. This talk will give a pragmatic overview of the motivations behind design patterns and will focus on applying a handful of the GOF patterns to example scenarios in Java and C#. A quick introduction to the role AOP plays in changing the patterns landscape will also be covered.

Lightweight Development Strategies by Bruce Tate

Based on the book Better, Faster, Lighter Java, this beginner to intermediate session will dive into philosophies for lightweight development. It's not a hardcore programming session, but we will talk about process, technologies like Spring, and design patterns like AOP and Dependency Injection. This philosophical session will talk about architectural philosophies rather than low-level programming issues.

Introduction to Spring by Bruce Tate

This session, for the Spring beginner, helps you: # Understand dependency injection and inversion of control # Know the meaning of lightweight containers and Spring # Understand the basic pieces of Spring # See core Spring modules in action, including Persistence, AOP, transactions. Attendees need not know anything about Spring. This session does talk about integration with core J2EE frameworks like JDBC and transactions.

Politics of Persistence by Bruce Tate

This session will help a Java developer choose a persistence framework. After the session, you will # Understand the core strengths and weaknesses of the main persistence frameworks in the Java space # Understand where marketing influences can impact persistence # Know what's going on behind the scenes to impact the persistence pictures # Answer questions about persistence frameworks that might not be mainstream

Ruby Persistence Strategies by Bruce Tate

Did you ever wonder how people from dynamic languages handle persistence? Here's your chance to see. This session will talk through some important Ruby persistence framework, including basic SQL-driven access (like JDBC), Active Record in Rails, and object relational mapping in Ruby via OG.

Beyond Java by Bruce Tate

All programming languages have a limited life span, and Java is no different. This is a philosophical session rather than a programming session. Sooner or later, Java will lose its leadership position. This session will explore Java's strengths and weaknesses. We'll try to understand whether conditions are ripe for alternatives to emerge, and what those alternatives may be.

Ruby for Java Programmers by Dave Thomas

Ruby recently enjoyed its tenth birthday. Instead of cake and candles, the community celebrated by releasing a wave of new libraries and frameworks that make Ruby programming even easier. This talk features some of the best of these, as we explore Ruby.

Herding Racehorses and Racing Sheep by Dave Thomas

Are you frustrated by experts who can't tell you what to do, or by junior team members who refuse to see the big picture? How can you best develop careers: both yours and those of your teammates and managers? How can we learn to apply experience more effectively, and why do the many approaches designed to tame complexity actually end up increasing it?

OpenSource Ecosystems by Dave Thomas

Open Source communities produce high quality software with little management and (typically) no pay. Most people looking at open source focus on using this software in their projects.

Ruby on Rails by Dave Thomas

The Ruby on Rails framework has exploded onto the scene over the last few months. Propelled by some genuine benefits, and fueled by a whole lot of controversy, Rails seems here to stay. So, is it a Java killer? (No.) Is it a great way to develop certain classes of web application? (Yes.) Does it really deliver the 10-fold increase in developer productivity that some have claimed? (It depends...)

Software Metrics and the Great Pyramid of Giza by David Bock

Most software engineers hate metrics... Why? Because we know the work we do is hard to quantify # any measurement of 'software engineering' is like trying to tell how tall someone is by how much they weigh... There may be some correlation, but there is so much deviation as to make the answer practically meaningless. As a result, we often see metrics used to justify improper conclusions. There are plenty of good metrics though, and plenty of ways to use them effectively.

Designing and Developing Pluggable Application Architectures by David Bock

Pluggable application architectures are everywhere. Applications like Photoshop, Eclipse and other IDEs, and even application servers are all examples of applications that allow other developers to 'install' new functionality. There are plenty of reasons for wanting to install new functionality into an application that is already developed and deployed... from dynamic upgrading to the creation of a 'component marketplace', where end users can purchase components with 'extra' capability.

Clean scalable builds with Maven by Dion Almaer

Our build systems have migrated from make to Ant. While Ant does a good job in many ways, is it the right tool for the job? This session talks about taking builds to the next level, looking at tools such as Maven to make your life easier.

How to be Groovy by Dion Almaer

What? Another programming language? Are you kidding me? That is what we often feel when something new comes around, and is something you may be feeling about Groovy. However, Groovy could fit a niche for you in your daily toil. It is the swiss army nice that Perl/Ruby are, yet lets you work in a more structured

way, and plays nice with the millions of lines of code already written on top of the Java Virtual Machine.

Enterprise AOP by Dion Almaer

Aspect-oriented programming (AOP) has become a hot topic for enterprise development, with recent news of support by IBM, JBoss, BEA, Oracle, Eclipse, and IntelliJ. Behind the news headlines, however, are critical questions: How real is AOP for the enterprise? What problems can it solve today? How does it apply to enterprise applications? How can one make an informed decision about trying to use AOP? What is the best adoption strategy? What are the long term possibilities for AOP in the enterprise? This sessions tries to tackle those questions.

Give the DB a break!: Performance and Scalability by Dion Almaer

What do we really mean by "performance" and "scalability"? This talk gets into the meat of problems which cause our applications to degrade. We will focus on issues such as problems caused by the database being a bottleneck for our application, and see how we can architect our solutions to bypass the issues, resulting in a solid system which scales with the increased load. Not only will we look at the factors, but I will delve into a couple of case studies to show how real world problems were solved!

Rules Engines by Dion Almaer

Rules engines are powerful beasts which allow you to program in a way in which you specific rules and facts, rather than a linear set of instructions. Learn about how you can use Rules Engines in Java development to take care of complicated problems.

Writing Secure Web Services (with Java and Axis) by Justin Gethland

Web Services are message-oriented. This means that any application intention (the need for security, for transactionality, for reliability, etc.) must be included in the message and not just assumed as external context. The WS-Security specifications are very advanced and currently being used in the wild to create robust, secure web services.

Principles of Service Oriented Architecture by Justin Gethland

SOA (Service Oriented Architectures) are the logical inheritor to the Enterprise Application Integration throne. As distributed applications begin to span more than just multiple servers, but multiple application servers, platforms, networks and transport protocols, the techniques for integrating the components together must change. Service Oriented Architectures are based on the notions of open messaging protocols, publishable interface contracts and shared standards.

Spring MVC by Justin Gethland

The Spring team, as in all things they do, have learned the valuable lessons of the past when introducing a Spring solution. Spring MVC is everything Struts should be, and more besides.

Spring Security with ACEGI by Justin Gethland

Spring offers developers a simpler, more robust method for configuring applications. These benefits extend to security through the ACEGI framework. ACEGI makes the otherwise daunting task of securing your application logical and straightforward. More importantly, through its support for single sign-on provision through Yale's CAS system and its ability to provide instance-level authorization, Spring extends the common security model of most J2EE apps beyond what they are traditionally capable of.

Advanced Hibernate by Justin Gethland

Hibernate is easy to get started with, but can sometimes be hard to make efficient or secure. In fact, the default settings for Hibernate create applications that will run slowly, cause unwanted round trips to the database, and may be more restrictive and/or permissive from a security standpoint than you would otherwise want.

Introduction to Hibernate by Justin Gethland

O/RM (Object/Relational Mapping) seeks to eliminate repetitive or tedious work enabling the CRUD (create, read, update, delete) that underlies most applications. Hibernate is a popular, open-source O/RM tool that uses reflection (instead of code generation, like EJB, or bytecode injection, like JDO) to manage your persistence layer. This session will introduce you to Hibernate. After an overview of common usage scenarios, including web and enterprise applications, we'll examine the basics of getting Hibernate running. We'll cover the mapping file format and syntax, including common relational mapping structures. Then, we'll examine the Hibernate API for interacting with the framework. Finally, we'll cover the common architectural decisions you'll have to make as you include this (or any other) O/RM framework.

Building Applications with the Spring Framework by Keith Donald

You'll see how to use Spring to assemble a complex system from a set of focused, loosely-coupled components. You'll experience through example how Spring enables agile development by allowing you to start simple, validate architectural choices early, and scale up as requirements demand.

Advanced Spring: What's New and What You Might Not Know About by Keith Donald

Spring 1.2 is out--Spring 1.3 is right on the horizon. As a broad, user-driven project with a large community, the newest releases offer a wealth of new features to be taken advantage of. This session focuses on demonstrating the most important, and how you can start leveraging them in your projects immediately.

J2EE Transaction Management Part 1 by Mark Richards

Although the EJB container isolates us from most of the complexities involving transaction management, there are still a number of things we need to be aware of when dealing with transactions within the J2EE container. Too often transaction management is an afterthought in the design and development process, which leads to applications that have problems with data integrity, data consistency, and overall stability and reliability. In this session we will explore the three transaction models that J2EE supports (Local, Programmatic, and Declarative), and discuss the advantages, disadvantages, and pitfalls within each of these models, when it makes sense to use each transaction model, and under what situations these models are appropriate and inappropriate. We will spend most of our time on the Declarative transaction model, otherwise known as container managed transactions (CMT). Within this model we will explore some common pitfalls and look at the best practices within this model. Through coding examples and real-world scenarios, you will learn how to properly handle exceptions, how to correctly use transaction attributes, and how the isolation level can affect transaction and application behavior. We will also discuss the problems encountered with Entity Bean data caching and data synchronization within the context of JTA transactions. This session is the first part of a 3 hour session.

Hibernate and J2EE Transaction Integration by Mark Richards

Hibernate is quickly becoming a viable alternative to the EJB Entity Bean and DAO persistence models. More and more applications are being written using Hibernate, and many existing applications are replacing legacy Entity Bean or DAO persistence models with Hibernate. Hibernate has it's own transaction manager and transaction API that is simple to configure and use for single resource, web-based applications. However, things get a bit more complicated when you consider using Hibernate as the persistence model for larger server-based J2EE Enterprise Applications. If you are planning on using Hibernate in conjunction with your existing app server, then you should attend this session. In this session we will take a look at how the Hibernate transaction model works and how to integrate Hibernate into the JTA transaction model within J2EE. Through this session you will learn where Hibernate used as-is will cause runtime exceptions when integrating with JTA declarative transactions. You will also see that we in fact cannot use the Hibernate Transaction API for JTA Declarative Transaction processing, and what steps are required to integrate with JTA given this fact. Also, you will learn how the Hibernate Transaction API violates the EJB 2.1 spec, and ways to get around these violations. We will end this session with a brief discussion about the application architecture considerations with integrating Hibernate as our persistence model for existing large-scale J2EE applications.

A Pragmatic Look at Agile Architecture by Mark Richards

Designing application and enterprise architectures is a complex process. We follow defined processes, create lots of attractive architecture diagrams, kill lots of trees producing hundreds of pages of architecture documentation, and yet we find that in many cases the architectures we design are not followed by developers or simply not understood by the stakeholders or development community. As a result, either the delivered software does not match the original architecture, or the architect works 25 hours a day to ensure that the software is in compliance with the original architecture. This is a case where Agile Architecture can help. Agile architecture is an architecture process that leverages Agile principles and applies them to the design of application and enterprise architectures. During this session you will learn the shortcomings of standard architecture processes and why they sometimes go wrong. We will then look at some Agile principles and see how we can apply these methods to simplify the architecture process and produce better architectures. We will also look at some pragmatic techniques for architecture design and diagramming that support Agile Architecture, and learn how ATAM (Architecture Trade-off Analysis Method) can be applied to the concept of Agile Architecture. Of course, no pragmatic session would be complete without a discussion of the issues and shortcomings of the Agile Architecture process itself, so we will explore where the Agile Architecture process sometimes fails to live up to its promise. There will be plenty of room for discussion in this session, so please bring your architecture pains (and successes) with you so we can engage in a lively discussion.

J2EE Transaction Management Part 2 by Mark Richards

This session is the second part of a 3 hour transaction management session. In this session we will explore some of the more advanced features of J2EE transaction management. We will pick up where we left off from the first session by taking a detailed look at XA and J2EE distributed transaction processing, and how to coordinate multiple resources within a single business transaction. Within the XA discussion you will learn what XA is, what the relationship is between JTA and XA, when you should use XA within J2EE applications, and how to enable JMS and DBMS resources to run under XA. We will also explore XA Drivers, and discuss the many issues surrounding XA Drivers within J2EE. We will then look at how to build an effective transaction design strategy by looking at three primary transaction strategy design patterns. We will review sample transaction code and settings within each pattern, and see how these patterns fit into different J2EE application architectures. We will end by looking at the steps for migrating from from a Local Transaction Model to a Declarative Transaction Model.

Introduction to Java Server Faces by Neal Ford

This session introduces experienced Java web developers to the new framework on the block and talks about the major features of JSF. It also compares JSF to other frameworks along the way.

Web Application Security Vulnerabilities by Neal Ford

This session highlights common mistakes made by web programmers, stating the problems and avoidance techniques.

Language-oriented Programming and Language Workbenches: Building Domain Languages atop Java by Neal Ford

This session shows how to use Java as the building block for domain-specific languages. It discusses the next revolution in programming: language-oriented programming and the nascent tools that support it.

Power Regular Expressions in Java by Neal Ford

This session shows how to fully exploit regular expressions in Java.

Advanced Enterprise Debugging Techniques by Neal Ford

Out, out, damn bugs! This session discusses techniques and tools for debugging enterprise applications.

Comparison of Java Web Frameworks by Neal Ford

This session provides an objective comparison of Struts, Tapestry, and JavaServerFaces.

Continuous Integration using CruiseControl and CVS by Paul Duvall

Continuous Integration (CI) is the process of continually building and testing your software under development. It is identified as a core XP practice, although it works with many software development processes.

Software Infrastructure Patterns by Paul Duvall

Design Patterns became part of the software development industry mainstream in the mid-1990s with the release of the Go4 Design Patterns book. Since then, architecture, design, and more recently, organizational patterns have become a part of our nomenclature. But, what about the software that helps us develop and deliver the software to our users: the software infrastructure?

Aspect-oriented refactoring: Taking refactoring to a new level by Ramnivas Laddad

Refactoring allows reorganizing code while preserving the external behavior, while AOP facilitates modularizing crosscutting concerns in a system through use of a new unit of modularity called aspect. Aspect-oriented refactoring synergistically combines these two techniques to refactor crosscutting elements. Individually, refactoring and AOP both share the high-level goal of creating systems that are easier to understand and maintain without requiring huge upfront design effort. A combination of the two # aspect-oriented refactoring # helps in reorganizing code corresponding to crosscutting concerns to further improve modularization that is easy to understand, highly consistent, and simple to change.

Introduction to Aspect-oriented programming with AspectJ by Ramnivas Laddad

Aspect Oriented Programming (AOP) enables modularizing implementation of crosscutting concerns that abound in practice: logging, tracing, dynamic profiling, error handling, service-level agreement, policy enforcement, pooling, caching, concurrency control, security, transaction management, business rules, and so forth. Traditional implementation of these concerns requires you to fuse their implementation with the core concern of a module. With AOP, you can implement each of the concerns in a separate module called aspect. The result of such modular implementation is simplified design, improved understandability, improved quality, reduced time to market, and expedited response to system requirement changes. Come to

this session and learn all about how AOP can help you simplify developing complex systems.

Design Pattern Modularization with AOP by Ramnivas Laddad

Design patterns # object oriented, concurrency control, and J2EE # all have certain crosscutting elements present. The obvious result of conventional implementation is unclear implementation that is tedious to implement and tough to change. Aspect-oriented programming (AOP) offers a way to simplify implementation of these design patterns. Further, AOP offers new design patterns of its own that allow for new ways of implementing functionalities.

Performance monitoring in J2EE applications by Ramnivas Laddad

J2EE has become the main new platform for enterprise application deployment. Good performance is an important requirement from the business viewpoint. Supporting this requirement needs application profiling during the development phases and performance monitoring after deploying the application. Come to this session to understand challenges and choice in monitoring J2EE applications.

Class Loading in Java: Building Dynamic Systems Without Pain by Stuart Halloway

(3 Hour Session) One of Java's greatest strengths is its flexible deployment model. In this session you will learn how Class Loaders facilitate deployment, and how to troubleshoot Java and J2EE Class Loading problems.

Cryptography for Programmers by Stuart Halloway

For centuries people have used crypto to build (and break) secure systems. Computers have only raised the pitch of conflict, providing enormous cryptographic power at commodity prices. Most programmers do not write their own crypto libraries, instead relying on the services of an operating system or virtual machine. But even with all this support, building secure systems is a daunting task.

Java Platform Security and JAAS by Stuart Halloway

The Java platform is built from the ground up with security in mind. This talk will introduce the security features of the J2SE, building quickly from the basic classes to realistic examples.

Good, Bad and Ugly of Java Generics by Venkat Subramaniam

Java introduced Generics in the 1.5 version (Java 5). What are the capabilities of Generics? How do you use it? Are there some gotchas in using it? In this example driven presentation, we will start at the basics of generics and look at its capabilities. We will then look at some of the under the hood details on generics implementation. We will then delve into the details of some of the changes to Java libraries to accommodate generics. Finally we will take a look at some restrictions and pitfalls that we need to be familiar with when it comes to practical and prudent use of generics.

Test First Development by Venkat Subramaniam

Do you know that unit testing is more of an act of design than verification? What are its benefits? How do we write effective tests? How does unit testing relate to evolutionary design? How does it help you with refactoring? When should you write your tests? What are the types of tests you could write? These are some of the questions that you would ask if you are interested in Unit Testing. What is a better way to learn than practicing it? In this session the attendees will participate in designing and developing a small yet full application. Instead of PowerPoint slides, you will learn from example. The code you help develop will be available for free download on the speaker's web site.

Prudent OO Design by Venkat Subramaniam

Is your code object-oriented? Developing with objects involves more than using languages like Java, C#, C++ or Smalltalk for that matter. From time to time, the OO paradigm stumps even expert developers. Agile programming becomes a mere act of hack if we code without knowing the OO principles. What are these principles # the ones that influence your design? In this presentation the speaker will present some of the challenges that are fundamental in nature. Then he will present OO Design principles and good practices for prudent development of OO code.

Agile Software Development by Venkat Subramaniam

You have probably worked on a few projects that have succeeded and then a few that have failed. What were the factors that influenced the success or failure of those projects? You want to develop a system that is robust, maintainable, within budget, of high quality and with fewer defects. How can you realize those goals? What steps, process, tools you can use or follow to achieve this. In this session, the speaker will present a number of approaches that lead to successful development. He will also present his personal experience with those in implementing software projects. Attendees are encourage and expected to present

their views on what has or has not worked for them.