Conference Chair: Ralph Johnson, University of Illinois

It's a pleasure to welcome you to OOPSLA 2005, the 20th annual conference on object oriented programming, systems, languages, and applications. OOPSLA is the premier forum for practitioners, researchers, and students in diverse disciplines whose common thread is object technology. From its inception, OOPSLA has served as an incubator for advanced technologies and practices. Dynamic compilation and optimization, Java, patterns, refactoring, aspect-oriented programming, UML, and agile methods (to name a few) all have OOPSLA roots.

OOPSLA 2005 continues that tradition. Researchers and practitioners from around the world have come to showcase their latest work. Presentations from invited speakers dovetail with technical papers, practitioner reports, expert panels, demonstrations, formal and informal educational symposia, workshops, and diverse tutorials from world-class lecturers. The popular Onward! track presents out-of-the-box thinking at the frontiers of computing. You can discuss late-breaking results with the researchers themselves at poster sessions, which culminate in the Fourth Annual SIGPLAN Student Research Competition. DesignFest® provides hands-on design experience in an expert-mentored environment. As always, the opportunities for mingling and professional networking are boundless.

In fact, there's so much to do and see at OOPSLA that planning a schedule can be daunting, particularly for newcomers. So whether this is your first OOPSLA or you're just feeling overwhelmed, be sure to join us Monday evening for the Newcomer Orientation right after the Welcome Reception. We'll answer your questions and give you tips on how to navigate the conference.

All these activities produce a unique environment for learning and collaboration. Take advantage. Catch up with your far-flung colleagues. Ask questions of the speakers. See the demos and interact with poster presenters. Don't be shy about investigating areas outside your expertise. OOPSLA's diversity means there are lots of opportunities for cross-pollination, and cross-pollination is a key to innovation.

Most of all, take what you learn here and apply it to your work-life. OOPSLA is all about impact, not just on the research community but on practitioners, educators, and students as well. It's our work as a community that gives rise to broad advancements in computing; no person or organization can do it alone. I believe every one of us can look forward to greater success at work, individually and collectively, through our experience at OOPSLA.

Thanks for attending, and enjoy the conference!

Monday Evening Activities

Welcome Reception
17:30-19:30
Terrace Pavilion and Poolside Area

Newcomer Orientation
19:30-20:00
Sunrise Room

Supporters

Gold Corporate Supporters

Since its inception in 1982, a singular vision — "The Network is the Computer" — has propelled **Sun Microsystems, Inc.** to its position as a leading provider of industrial-strength hardware, software and services that make the Net work. Sun can be found in more than 100 countries and on the World Wide Web at http://sun.com.

Sun Microsystems Laboratories (Sun Labs) is Sun's applied research and advanced development arm. Sun Labs researchers work on projects that play a significant role in the evolution of technology and society - projects that have the power to change everything - including asynchronous and high-speed circuits, optical interconnects, third-generation web technologies, sensors, network scaling, SunRay and Java technologies. Since its inception, Sun Labs has been responsible for many of Sun's technology advancements and inventions, maintaining one of the highest rates of technology transfer in the industry. For more information about Sun Labs, please visit: www.research.sun.com.

Founded in 1975, **Microsoft** is the worldwide leader in software, services and Internet technologies for personal and business computing. The company offers a wide range of products and services designed to empower people through great software — any time, any place and on any device. Microsoft Visual Studio 2005 is the comprehensive tool set for rapidly building and integrating Web services, Microsoft Windows-based applications, and Web solutions. For more information, please visit http://www.microsoft.com.

IBM is the world's largest information technology company with 80 years of leadership in helping businesses innovate. We create tools and technologies that will enable the continued evolution of computing and computing services over the network. Our work across many disciplines is often done in concert with our colleagues in academic and government research centers, as well as "in the marketplace" with customers who provide us with challenging research problems.

IBM Software offers the widest range of applications, middleware and operating systems for all types of computing platforms, allowing customers to take full advantage of the new era of on-demand e-business. The fastest way to get more information about IBM software is through: http://www.software.ibm.com.

IBM Research is the world's largest information technology research organization, with more than 3,000 scientists and engineers at eight labs in six countries. IBM has produced more research breakthroughs than any other company in the IT industry. For more information on IBM Research, visit http://www.research.ibm.com.

Bronze Corporate Supporters

Google is a global technology leader focused on improving the way people connect with information. Google's innovations in web search and advertising have made its website a top Internet destination and its brand one of the most recognized in the world. Google maintains the world's largest online index of websites and other content, and Google makes this information freely available to anyone with an Internet connection. Google's automated search technology helps people obtain nearly instant access to relevant information from it vast online index. For more information, visit www.google.com.

Addison-Wesley and **Prentice Hall** are the leading publishers of high-quality and timely resources for programmers, managers, engineers, and system administrators. Our readers rely on us to provide up-to-date information on the latest technologies, as well as new approaches to current technologies. Written by the industry's most respected experts, our educational materials teach technology professionals essential new skills and timesaving techniques to help them do their jobs more effectively. Addison-Wesley and Prentice Hall are part of the Pearson Technology Group. As a division of Pearson Education, our readers and authors benefit from our global resources. For more information, please visit www.awprofessional.com and www.phptr.com.

Corporate Supporters

Springer

O'Reilly

Table of Contents

Sunday

	8:30-12:00	13:30-17:00
DesignFest	*San Diego Room*	
	San Diego Room	San Diego Room
Tutorials *(lunch is included in the Town & Country Room)*	**17 Essential Object-Oriented Analysis and Design** Jill Aden, Joseph Brennan *Pacific Salon 5* **18 Software Factories: Assembling Applications with Patterns, Models, Frameworks and Tools** Jack Greenfield, Steve Cook *Pacific Salon 3*	
	GP1 How has the arts, sports or life stimulated, inspired and informed your work in computer science? George Platts *Terrace Pavilion and Poolside* **1 Patterns in Functional Programming** Jeremy Gibbons *Pacific Salon 6* **2 Tuning Your Methodology to You** Alistair Cockburn *Pacific Salon 2* **3 Enterprise Aspect-Oriented Programming with AspectJ** Ronald Bodkin *Pacific Salon 7* **4 Code Smells** Ken Scott-Hlebek *Sunset* **5 Adding Software Testing to Programming Assignments** Stephen Edwards *Pacific Salon 4* **6 Working Effectively with Legacy Code** Michael Feathers *Sunrise* **7 Generative Software Development** Krzysztof Czarnecki *Esquire* **8 Adaptive Object-Model Architecture: How to Build Systems That Can Dynamically Adapt to Changing Requirements** Joseph Yoder *Pacific Salon 1*	**GP2 Querdenking: Can creativity be taught?** George Platts *Terrace Pavilion and Poolside* **9 Steps to an Agile Frame of Mind - the First Three Hours** Alistair Cockburn *Pacific Salon 2* **10 The Six Million Dollar Customer** Angela Martin, Robert Biddle, James Noble *Pacific Salon 6* **11 Teaching Java: An Eventful Approach** Kim Bruce *Pacific Salon 4* **12 Use Cases Are Early Aspects** Ivar Jacobson, Pan-Wei Ng *Sunrise* **13 The Common Lisp Object System: Generic Functions and Metaobject Protocol** Pascal Costanza *Pacific Salon 7* **14 A Tour of Responsiblity-Driven Design** Rebecca Wirfs-Brock *Pacific Salon 1* **15 Storytest-Driven Development** Ken Scott-Hlebek *Sunset* **16 Hands On AspectJ Development for Enterprise** Ron Bodkin *Esquire*
Wiki Symposium		**WikiSpam Workshop** *Eaton*
Workshops	**Apprenticeship Pedagogical Patterns** *Royal Palm Salon 6* **Best Practices for Model Driven Software Development** *Royal Palm Salon 5* **Beyond the Project Myth - Agile Development and Product Environments** *Brittany* **Building Software for Pervasive Computing** *Royal Palm Salon 4* **Early Aspects** *Royal Palm Salon 3* **Eclipse Technology eXchange (ETX) 2005--Day 1** *Golden West Room* **Extravagaria III: Hunting Creativity** *Royal Palm Salon 2* **Fostering Software Reliability in an Increasingly Hostile World** *Fairfield* **Fourth "Killer Examples" for Design Patterns and Objects First Workshop** *Clarendon* **Library-centric software design** *Dover* **Synchronization and concurrency in object-oriented languages (SCOOL)** *California Room* **Third Workshop on Method Engineering for Object-Oriented and Component-Based Development** *Royal Palm Salon 1*	
Evening	**IBM Reception: Eclipse Technology Exchange Poster Session** *18:30-21:00 Terrace Pavilion* *(light refreshments)*	

Monday

	8:30-12:00	13:30-17:00
Doctoral Symposium	*Stratford*	
Educators' Symposium	*San Diego Room*	
Tutorials *(lunch is included in the Town & Country Room)*	**35 Mastering UML with Stable Software Patterns** Mohamed Fayad, Haitham Hamza *Esquire* **36 Pattern-Oriented Software Architecture: Patterns for Concurrent and Distributed Systems** Douglas Schmidt *Pacific Salon 1*	
	19 Java Reflection Ira Forman, Nate Forman *Pacific Salon 4* **20 Introduction to Concurrent Programming in Java 5.0** David Holmes, Brian Goetz *Pacific Salon 5* **21 Agile Requirements: Tailoring the Functional Requirements Specification Process to Improve Agility** Jennitta Andrea, Geoff Hardy *Sunset* **22 Extending the Standard Widget Toolkit - how to create your own widgets** Veronika Irvine, Steve Northover *Pacific Salon 3* **23 Foundations of object-oriented languages: Types and Language Design** Kim Bruce *Pacific Salon 2* **24 Use-case patterns and blueprints** Gunnar Overgaard, Karin Palmkvist *Pacific Salon 7* **25 Effective Interface Design: Seven Recommendations for Improving the Design of Interfaces in Code** Kevlin Henney *Sunrise* **26 Merciless Refactoring with Eclipse** Martin Lippert *Pacific Salon 6*	**27 Functional Acceptance Testing: The Essentials** Jennitta Andrea *Pacific Salon 7* **28 Programming without a Call Stack - Event-Driven Architectures in Action** Gregor Hohpe *Sunset* **29 Find Your Voice** Gail E. Harris *Pacific Salon 6* **30 Effective Concurrent Programming with Java 5.0** David Holmes, Tim Peierls, Brian Goetz, Joe Bowbeer *Pacific Salon 3* **31 STL Patterns: A Design Language of Generic Programming** Kevlin Henney *Pacific Salon 5* **32 Domain-Driven Design: Putting the Model to Work** Eric Evans *Sunrise* **33 Large-Scale Software Architecture: A Practical Guide Using UML** Jeff Garland, Richard Anthony *Pacific Salon 2* **34 What the Eclipse Modeling Framework (EMF) is and how you can use it in your applications.** Marcelo Paternostro *Pacific Salon 4*
Wiki Symposium	*Sessions in California Room, Ascot Room and Terrace Pavilion & Poolside*	
Workshops	**Croquet: A Platform for Collaboration** *Eaton* **Eclipse Technology eXchange (ETX) 2005--Day 2** *Golden West Room* **Fourth International Workshop on Agent-Based Methodologies** *Royal Palm Salon 2* **International Workshop on Software Factories** *Royal Palm Salon 6* **Java Technologies for Real-Time and Embedded Systems** *Royal Palm Salon 4* **Multiparadigm Programming in Object-Oriented Languages** *Fairfield* **Scrapheap Challenge—A Workshop in Post-Modern Programming** *Clarendon* **The 5th OOPSLA Workshop on Domain-Specific Modeling** *Royal Palm Salon 1* **Third Int'l Workshop on SOA and Web Services Best-practices** *Royal Palm Salon 3*	
Evening *(light refreshments at receptions)*	**Welcome Reception** and **Posters** *17:30-19:30 Terrace Pavilion and Poolside Area* **Wiki Symposium Welcome Reception** *17:30-19:30 Poolside Area near Trellises Rest.* **Newcomer Orientation** *19:30-20:00 Sunrise Room* **Birds of a Feather (BoF)** *See Information Desk*	

Tuesday

	8:30-10:00	10:30-12:00
Keynotes & Invited Speakers	**OOPSLA 2005 Keynote: Creativity** Robert Hass *Town & Country Room*	**Wikipedia in the Free Culture Revolution** Jimmy Wales *Golden West Room*
Research Papers		**Type Types** *San Diego* Associated Types and Constraint Propagation for Mainstream Object-Oriented Generics Generalized Algebraic Data Types and Object-Oriented Programming Scalable Component Abstractions
Onward!		**Onward! Papers** *Town & Country Room* Subtext: Uncovering the Simplicity of Programming X10: An Object-Oriented Approach to Non-Uniform Cluster Computing
Essays & Practitioner Reports		**A Simple Model of Agile Software Processes - or - Extreme Programming Annealed** Glenn Vanderburg *Sunrise*
Panels		
Demonstrations		*11:00-11:45* *Courtyard (rooms A, B & C)* *12:00-12:45* *Courtyard (rooms A, B & C)*
Tutorials (13:30 – 17:00)	****You can still sign up for tutorials at the registration desk **** *(a box lunch is included)*	
Dynamic Language Symposium	**Welcome** (1/2 hr) *Sunset* **Objects as Software Services** (1 hr)	**Language Mechanisms and Extensions** *Sunset*
Wiki Symposium	*Symposium attendees are invited to the OOPSLA 2005 Keynote* *Town & Country Room*	*See Invited Speaker* *Golden West Room*
Other Events		**Instant Arts School Experience - just add people** *Courtyard*

See Page 94 for lunch ideas

Tuesday

13:30-15:00	15:30-17:00	Evening
Why Programming is a Good Medium for Expressing Poorly Understood and Sloppily Formulated Ideas Gerald Jay Sussman *Golden West Room*		
Analysis Analyzed *San Diego* Demand-Driven Points-to Analysis for Java Deriving Object Typestates in the Presence of Inter-Object References Micro Patterns in Java Code	**Archetypal Architectures** *San Diego* ArchMatE: From Architectural Styles to Object-Oriented Models through Exploratory Tool Support Modeling Architectural Patterns Using Architectural Primitives Parametric Polymorphism for Software Component Architectures Using Dependency Models to Manage Complex Software Architecture	**FlashBoF** *(17:00-18:30)* *Royal Palm Salon 1 & 2*
Onward Presentations 1: Exploration *Town & Country Room*	**Breakthrough Ideas** *Town & Country Room*	**Munch, Muse, Mooch, Mutter, Mix and Mingle at the Mall** *(18:00-19:30)* *Fashion Valley Mall Food Court* *(Typical meals range from $5.00 to $10.00)*
	Practice and Principles *Golden West Room* Legacy System Exorcism by Pareto's Principle Estimating Software Based on Use Case Points Finding the Forest in the Trees	**Sun JCP Dessert Reception** *(19:00-20:00)* *Town & Country Room* *Celebrating 10 years of Java, join us for sweets and coffee.*
Aspects: Passing Fad or New Foundation? *Sunrise*		
15:00-15:45 *Courtyard (rooms A, B & C)*	*16:00-16:45* *Courtyard (rooms A, B & C)*	
WT1 Getting Started with Wikis Sunir Shah *Royal Palm Salon 3* **WT3 Using Wikis in Software Development** Jack Bolles *Royal Palm Salon 4*	**37 Organizational Patterns: Beyond Agility to Effectiveness** Neil B. Harrison, James O. Coplien *Royal Palm Salon 1 & 2* **38 PHP/MySQL for Community Programming** Ghica van Emde Boas *Royal Palm Salon 6* **39 First-Aid Clinic for Change Agents** Linda Rising, Mary Lynn Manns *Stratford*	**A 20th OOPSLA Retrospective** *(20:00-21:00)* *Town & Country Room*
See Invited Speakers *Golden West Room*	**Common Lisp: a dynamic language for a dynamic world** (1 hr) *Sunset* **Language Reasoning** (1 hr)	**I Have Nothing to Declare but My Genius** *(17:30)* *Sunset*
Workshops in Esquire, Dover and Royal Palm Salon 5 *(see also Tutorials)*		
	Instant Arts School Experience - just add people *Courtyard*	

Posters are on display from 10:00 to 17:00 in the Terrace Pavilion

Wednesday

	8:30-10:00	10:30-12:00
Keynotes & Invited Speakers	**Onward! Keynote: The End of Users** Mary Beth Rosson *Town & Country Room*	
Research Papers		**Language Lingo** *San Diego Room* Classbox/J: Controlling the Scope of Change in Java Interaction-Based Programming with Classages Javari: Adding Reference Immutability to Java
Onward!	*See Invited Speakers* *Town & Country Room*	**Onward Presentations 2: Ambient and Organic** *Town & Country Room*
Practitioner Reports		**Language and Reality** *Golden West Room* Honey, I shrunk the Types—How Behavioral Types lose relevance on the edges of OO Applications Removing duplication from java.io: a case study using Traits Arithmetic with Measurements on Dynamically Typed Object Oriented Languages
Panels		**Fostering Software Robustness in an Increasingly Hostile World** *Sunrise*
Demonstrations		*11:00-11:45* *Courtyard (rooms A, B & C)* *12:00-12:45* *Courtyard (rooms A, B & C)*
DesignFest		*Sunset (to 17:00)*
Tutorials (13:30 – 17:00)	****You can still sign up for tutorials at the registration desk **** *(a box lunch is included)*	
Other Events		**Student Research Finalist Presentations** *Royal Palm 1 & 2* **Instant Arts School Experience - just add people** *Courtyard*

See Page 94 for lunch ideas

Wednesday

13:30-15:00	15:30-17:00	Evening
	Designing Croquet's TeaTime - A Real-time, Temporal Environment for Active Object Cooperation David P. Reed *Golden West Room*	
Adaptation Adapted *San Diego Room* Fine-Grained Interoperability through Mirrors and Contracts Pluggable AOP: Designing Aspect Mechanisms for Third-party Composition Refactoring Support for Class Library Migration	**Machine Machinery** *San Diego Room* Automating Vertical Profiling Improving Virtual Machine Performance Using a Cross-Run Repository Quantifying the Performance of Garbage Collection vs. Explicit Memory Management Runtime Specialization With Optimistic Heap Analysis	
Working with Vision *Town & Country Room*	**Onward Presentations 3: Structure and Semantics** *Town & Country Room*	**FlashBoF** *(17:00-18:30)* *Sunset*
		A Visit to the San Diego Zoo *(19:00-23:00)* *Dinner included.* *Buses will be outside the Atlas Foyer (near registration).*
The Agile Panel *Golden West Room*		
14:00-14:45 *Courtyard (rooms A, B & C)* *15:00-15:45* *Courtyard (rooms A, B & C)*		
Dover and Stratford Rooms		
40 Agile Software Development in the Large Jutta Eckstein *Royal Palm Salon 1 & 2* **41 DocIt!: Agile Documentation of Object-oriented Frameworks** Ademar Aguiar *Royal Palm Salon 3* **42 Robust Communications Software** Greg Utas *Royal Palm Salon 4*	**43 Creating and Protecting Software Intellectual Property Rights** Dion Messer *Royal Palm Salon 5* **44 Making RUP Agile** Michael Hirsch *Esquire* **45 The Eclipse Debug Framework** Bjorn Freeman-Benson, Darin Wright *Royal Palm Salon 6*	
Instant Arts School Experience - just add people *Courtyard*		

Posters are on display from 10:00 to 17:00 in the Terrace Pavilion

Thursday

	8:30-10:00	10:30-12:00
Keynotes & Invited Speakers	**Finding Good Design** Martin Fowler *Town & Country Room*	
Research Papers	**Tracing Traces** *San Diego Room* Adding Trace Matching with Free Variables to AspectJ Finding Application Errors Using PQL: A Program Query Language Relational Queries Over Program Traces	**Concurrency II Concurrency** *San Diego Room* Formalising Java RMI with Explicit Code Mobility Lifting Sequential Graph Algorithms for Distributed-Memory Parallel Computation Safe Futures for Java
Onward!		**Onward! Films** *Town & Country Room*
Practitioner Reports		
Panels		**Echoes: Structured Design and Modern Software Practices** *10:30-12:00* *Golden West Room* **Living With Legacy: Love It or Leave It?** *12:00-13:30* *Golden West Room*
Demonstrations		*11:00-11:45* *Courtyard (rooms A, B & C)* *12:00-12:45* *Courtyard (rooms A, B & C)*
Workshops	**MVCDC 2: Managing Variabilities consistently in Design and Code** *Esquire* *(all day)*	
Tutorials *(a box lunch is included)*	**53 An Introduction to Requirements Engineering** *(all day)* Brian Berenbach *Royal Palm Salon 3*	
	46 Models and Aspects: How to use MDSD and AOSD together Markus Voelter, Martin Lippert *Royal Palm Salon 5* **47 Challenges in Object-Relational Mapping** Alan Knight *Royal Palm Salon 6*	**48 Building Service-Oriented Architectures with Web Services** Olaf Zimmermann, Mark Tomlinson *Royal Palm Salon 1 & 2* **49 Agile User Experience Design** Jeff Patton *Royal Palm Salon 4*
Other Events		**Instant Arts School Experience - just add people** *Courtyard*

See Page 94 for lunch ideas

Thursday

<table>
<tr><th>13:30-15:00</th><th>15:30-17:00</th><th>Farewell!</th></tr>
<tr><td></td><td>On Creating a Handbook of Software Architecture
Grady Booch Town & Country Room</td><td rowspan="11">Ice Cream Social and Invitation to OOPSLA 2006
(17:00)
Terrace Pavilion and Poolside</td></tr>
<tr><td>Exceptional Exceptions San Degio Room
Combining the Robustness of Checked Exceptions with the Flexibility of Unchecked Exceptions using Anchored Exception Declarations
Incrementalization Across Object Abstraction
PolyD: A Flexible Dispatching Framework</td><td></td></tr>
<tr><td>Panel: Yoshimi Battles the Pink Robots [Programmers v. Users: The Culture War]
Town & Country Room</td><td></td></tr>
<tr><td>Plans and Results Golden West Room
Service-Oriented Architecture and Business Process Choreography in an Order Management Scenario: Rationale, Concepts, Lessons Learned
Agility vs. Stability at a Successful Startup
Using Predicate Fields in a Highly Flexible Industrial Control System</td><td></td></tr>
<tr><td>See Onward! Town & Country Room</td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td colspan="2">Continuation of MVCDC 2: Managing Variabilities consistently in Design and Code Esquire</td></tr>
<tr><td colspan="2">Continuation of An Introduction to Requirements Engineering Royal Palm Salon 3</td></tr>
<tr><td colspan="2">50 The C# Programming Language
Mads Torgersen Royal Palm Salon 4
51 An Architects Guide to Enterprise Integration with J2EE and .NET
Ian Gorton, Anna Liu Royal Palm Salon 1 & 2
52 Pattern Languages Hands-On
Maria Kavanagh, Alan O'Callaghan Royal Palm Salon 5</td></tr>
<tr><td>Lightning Talks Sunrise
Instant Arts School Experience - just add people Courtyard</td><td></td></tr>
</table>

Posters are on display from 10:00 to 14:00 in the Terrace Pavilion

Invited Speakers

Chair: Richard P. Gabriel, Sun Microsystems, Inc.

```
his smile will be yours
I'll tell him

drive on
anyways
drive   on
```

Keynotes & Invited Speakers present new, different, or otherwise interesting perspectives on topics of broad and general interest. Speakers are invited by the Conference Committee based on the informal theme of the conference and the interests of the committee members. Good keynotes intrigue, provide new points of view, or focus attention on a problem or technology. Ideally a keynote encompasses the conference and centers the conversations and dialog that will go on between sessions, over meals, and late into the night. A conference is a heightened experience, and keynotes cause its participants to face a particular direction—and to face each other. Invited talks are more narrowly focused on a research area, a technology, or a problem area.

The opening keynote will be presented by Robert Hass. Robert Hass was the Poet Laureate of the United States from 1995 to 1997. He won't tell us what it will be about, but trust me, he'll be great. Bob was one of the first poets who critiqued my poetry, and he was amazed—jaw-dropped amazed—at how awful I was at reading my own work aloud. The Onward! keynote will be presented by Mary Beth Rosson. She was general chair in Minneapolis in 2000. You might remember Mary Beth throwing her hat in the air as part of a video/live intro to OOPSLA. That was the year I read my Mob Software essay at a plenary session on the last day filled with people who were amazed—jaw-dropped amazed—at my outfit (an agbada for those who are still wondering) and the fact I was reading my talk. I did this to try to prove to myself that Bob Hass was wrong and that I could read my own work aloud. Martin Fowler will present the third Keynote. As he says himself, Martin is an author, speaker, consultant and general loud-mouth on software development. His talks are fun and informative, and I find I learn something every time I hear him speak. Gerald Jay Sussman will deliver an invited talk. He is the Matsushita Professor of Electrical Engineering at the Massachusetts Institute of Technology. I have known Gerry since the early 1970s. I think he is one of the smartest, most creative people living today. Jimmy Wales is the inventor of Wikipedia and will deliver an Invited talk as part of the 2005 International Symposium on Wikis. I don't know too much about him, but Wikipedia is a tipping point for the Web. Grady Booch is, well, Grady Booch. He will present preliminary results of his Handbook of Software Architecture project. David Reed is sometimes called the / in TCP/IP. He invented the end-to-end argument. This makes him a legend. Now he's working on Croquet, and that's what he'll talk about.

OOPSLA 2005 Keynote: Creativity

Tuesday, 8:30-10:00 ***Town & Country Room***

Robert Hass, Chancellor of The Academy of American Poets

In his role as United States Poet Laureate, Robert Hass spent two years battling American illiteracy, armed with the mantra, "imagination makes communities." He crisscrossed the country speaking at Rotary Club meetings, raising money to organize conferences such as "Watershed," which brought together noted novelists, poets, and storytellers to talk about writing, nature, and community. For Hass, everything is connected. When he works to heighten literacy, he is also working to promote awareness about the environment. Hass believes that natural beauty must be tended to and that caring for a place means knowing it intimately. Poets, especially, need to pay constant attention to the interaction of mind and environment. And when he is talking about poetry itself, whether Matsuo Basho's or Elizabeth Bishop's, Hass is both spontaneous and original, offering poetic insights that cannot be found in any textbook.

Robert Hass has published many books of poetry including Field Guide, Praise, Human Wishes, and Sun Under Wood, as well as a book of essays on poetry, Twentieth Century Pleasures. Hass translated many of the works of Nobel Prize-winning Polish poet, Czeslaw Milosz, and he edited Selected Poems: 1954-1986 by Thomas Transtromer, The Essential Haiku: Versions of Basho, Buson, and Issa, and Poet's Choice: Poems for Everyday Life. Robert Hass's deep commitment to environmental issues led him to found River of Words (ROW), an organization that promotes environmental and arts education in affiliation with the Library of Congress Center for the Book. Hass is chairman of ROW's board of directors, and judges their annual international environmental poetry and art contest for youth. He is also a board member of International Rivers Network. Robert Hass was chosen as Educator of the Year by the North American Association on Environmental Education. Robert Hass was the guest editor of the 2001 edition of Best American Poetry.

Awarded a MacArthur Fellowship, twice the National Book Critics Circle Award (in 1984 and 1997), and the Yale Series of Younger Poets in 1973, Robert Hass is a professor of English at UC Berkeley.

Wikipedia in the Free Culture Revolution

Tuesday, 10:30-12:00 ***Golden West Room***

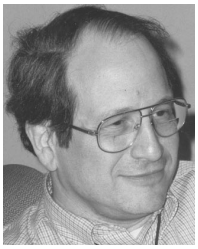
Jimmy Wales, Wikimedia Foundation

Wikipedia is one of the shining leaders of the free culture revolution taking place on the Internet. Wales will discuss how Wikipedia operates, what lessons can be learned for future projects, and what the future holds for free culture generally.

Jimmy "Jimbo" Wales is the founder of Wikipedia.org, the free encyclopedia project, and Wikicities.com, which extends the social concepts of Wikipedia into new areas. Jimmy was formerly a futures and options trader in Chicago, and currently travels the world evangelizing the success of Wikipedia and the importance of free culture. When not traveling, Jimmy lives in Florida with his wife and daughter.

Why Programming is a Good Medium for Expressing Poorly Understood and Sloppily Formulated Ideas

Tuesday, 13:30-15:00 ***Golden West Room***



Gerald Jay Sussman, Professor, Massachusetts Institute of Technology

I have stolen my title from the title of a paper given by Marvin Minsky in the 1960s, because it most effectively expresses what I will try to convey in this talk.

We have been programming universal computers for about 50 years. Programming provides us with new tools to express ourselves. We now have intellectual tools to describe "how to" as well as "what is". This is a profound transformation: it is a revolution in the way we think and in the way we express what we think.

For example, one often hears a student or teacher complain that the student knows the "theory" of some subject but cannot effectively solve problems. We should not be surprised: the student has no formal way to learn technique. We expect the student to learn to solve problems by an inefficient process: the student watches the teacher solve a few problems, hoping to abstract the general procedures from the teacher's behavior on particular examples. The student is never given any instructions on how to abstract from examples, nor is the student given any language for expressing what has been learned. It is hard to learn what one cannot express. But now we can express it!

Expressing methodology in a computer language forces it to be unambiguous and computationally effective. The task of formulating a method as a computer-executable program and debugging that program is a powerful exercise in the learning process. The programmer expresses his/her poorly understood or sloppily formulated idea in a precise way, so that it becomes clear what is poorly understood or sloppily formulated. Also, once formalized procedurally, a mathematical idea becomes a tool that can be used directly to compute results.

I will defend this viewpoint with examples and demonstrations from electrical engineering and from classical mechanics.

Gerald Jay Sussman is the Matsushita Professor of Electrical Engineering at the Massachusetts Institute of Technology. He received the S.B. and the Ph.D. degrees in mathematics from the Massachusetts Institute of Technology in 1968 and 1973, respectively. He has been involved in artificial intelligence research at M.I.T. since 1964. His research has centered on understanding the problem-solving strategies used by scientists and engineers, with the goals of automating parts of the process and formalizing it to provide more effective methods of science and engineering education. Sussman has also worked in computer languages, in computer architecture and in VLSI design.

Sussman is a fellow of the Institute of Electrical and Electronics Engineers (IEEE). He is a member of the National Academy of Engineering (NAE), a fellow of the American Association for the Advancement of Science (AAAS), a fellow of the American Association for Artificial Intelligence (AAAI), a fellow of the Association for Computing Machinery (ACM), a fellow of the American Academy of Arts and Sciences, and a fellow of the New York Academy of Sciences (NYAS). He is also a bonded locksmith, a life member of the American Watchmakers-Clockmakers Institute (AWI), a member of the Massachusetts Watchmakers-Clockmakers Association, a member of the Amateur Telescope Makers of Boston (ATMOB), and a member of the American Radio Relay League (ARRL).

Onward! Keynote: The End of Users

Wednesday, 8:30-10:00 ***Town & Country Room***

Mary Beth Rosson, Professor, Pennsylvania State University

Over the past 20 years, user interface designers and usability engineers have studied and refined human-computer interaction techniques with the goal of improving people's productivity and experience. But the target of these efforts "the end-user" is fast becoming a thing of the past. Many people now construct software on their own, building artifacts that range from email filters to spreadsheet simulations to interactive web applications. These individuals are use-developers: they build ad hoc solutions to everyday computing needs.

Will use-developers help to resolve the software crisis? Given the right tools, people and groups may be able to rapidly develop custom solutions to many context-specific computing requirements, eliminating the wait for IT professionals to analyze and engineer a solution. Or are these individuals a danger to society? Use-developers are informal programmers with no training in software construction methods or computing paradigms. They have little intrinsic motivation to test their products for even basic concerns like correctness or safety. In this talk I argue that the transformation of end-user to use-developer is well underway and discuss the prospects for maximizing the benefits to society while addressing the risks.

Mary Beth Rosson is Professor of Information Sciences and Technology at The Pennsylvania State University. She received a PhD in experimental psychology in 1982 from the University of Texas. Prior to joining the new School of Information Sciences and Technology at Penn State in 2003, she was a professor of computer science at Virginia Tech for 10 years and a research staff member and manager at IBM's T. J. Watson Research Center for 11 years. Rosson was among the earliest researchers to study the psychological issues associated with the object-oriented paradigm, and spent many years developing and evaluating object-oriented tools and training for professional programmers. One of her abiding interests has been the interplay between the concerns of human-computer interaction and software engineering. Recently she has been studying the tools and practices of end-user developers in educational and general business contexts.

Rosson has participated in the OOPSLA conference in many capacities, as a member of the technical program committee as well as the conference committee; she introduced the OOPSLA Educators' Symposium in 1992, the Doctoral Consortium in 1994, and served as General Chair for OOPSLA 2000. She is also active in the ACM SIGCHI community, where she is General Chair for CHI 2007. Rosson is author of Usability Engineering: Scenario-Based Development of Human-Computer Interaction (Morgan Kaufmann, 2002) as well as numerous articles, book chapters, and tutorials. More information is available at http://ist.psu.edu/rosson.

Invited Speakers

Designing Croquet's TeaTime - A Real-time, Temporal Environment for Active Object Cooperation **Wednesday, 15:30-17:00** ***Golden West Room***

David P. Reed, Co-architect of Croquet

Underlying Croquet is an object-oriented semantics based on active objects that have the capability of temporal reflection. That is, each object is aware and in direct control of its behavior in time. Croquet also directly supports replication of computation, allowing computation to be moved close to the point of interaction on demand, while maintaining a consistent view of behaviors that can scale to include thousands of nodes.

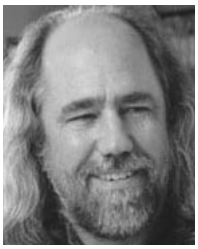
In the talk, we'll highlight the main concepts of TeaTime, which provides that semantic model, and also talk about some of the interesting implementation issues involved in realizing the TeaTime semantics.

Finding Good Design **Thursday, 8:30-10:00** ***Town & Country Room***

Martin Fowler, Chief Scientist, ThoughtWorks

Martin Fowler was best described by Brian Foote as "an intellectual jackal with good taste in carrion". He's not come up with great languages or tools, built major companies or found academic success. He's an author who has struggled with understanding what good design might be and how to communicate it. His books on patterns, refactoring, UML, and agile development reflect this question and his struggles to find an answer. He hasn't succeeded yet, but is happy to share his current position, lost in a maze of twisty objects, all alike.

On Creating a Handbook of Software Architecture **Thursday, 15:30-17:00** ***Town & Country Room***



Grady Booch, Free Radical, IBM

It is a sign of maturity for any given engineering discipline when we can name, study, and apply the patterns relevant to that domain. In civil engineering, chemical engineering, mechanical engineering, electrical engineering, and now even genomic engineering, there exist libraries of common patterns that have proven themselves useful in practice. Unfortunately, no such architectural reference yet exists for software-intensive systems. Although the patterns community has pioneered the vocabulary of design patterns through the work of the Hillside Group and the Gang of Four, our industry has no parallel to the architecture handbooks found in more mature design disciplines.

Following the work of Bruce Anderson, who over a decade ago conducted a series of workshops at OOPSLA, I've begun an effort to fill this void in software engineering by codifying the architecture of a large collection of interesting software-intensive systems, presenting them in a manner that exposes their essential patterns and that permits comparison across domains and architectural styles.

In this presentation, we'll examine the nature of architectural patterns and the process of conducting architectural digs to harvest them, and then examine a few of the systems studied thus far.

Grady is recognized internationally for his innovative work on software architecture, software engineering, and modeling. A renowned visionary, he has devoted his life's work to improving the effectiveness of software developers worldwide. Grady served as Chief Scientist of Rational Software Corporation since its founding in 1981 and continues to serve in that role within IBM. Grady is one of the original authors of the Unified Modeling Language (UML) and was also one of the original developers of several of Rational's products. Grady has served as architect and architectural mentor for numerous complex software-intensive projects around the world in just about every domain imaginable.

Grady is the author of six best-selling books, including the UML Users Guide and the seminal Object-Oriented Analysis with Applications, and has published several hundred articles on software engineering, including papers published in the early '80s that originated the term and practice of object-oriented design.

Grady is a member of the Association for Computing Machinery (ACM), the Institute of Electrical and Electronics Engineers (IEEE), the American Association for the Advancement of Science (AAAS), and Computer Professionals for Social Responsibility (CPSR). He is an IBM Fellow, an ACM Fellow, a World Technology Network Fellow, a Software Development Forum Visionary, and generally just a really nice and gentle fellow. Grady was a founding board member of the Agile Alliance, the Hillside Group, the Worldwide Institute of Software Architects, and now also serves on the board of the International Association of Software Architects. He also serves on the boards of Northface University and the Iliff School of Theology.

Grady received his bachelor of science from the United States Air Force Academy in 1977 and his master of science in electrical engineering from the University of California at Santa Barbara in 1979.

Evening Events

OOPSLA 2005 provides the opportunity to mingle with the recognized leaders in object technology in a more casual setting through a number of social activities designed just for that purpose.

Welcome Reception (light refreshments) **Monday, 17:30-19:30** ***Terrace Pavilion & Poolside Area***

The Welcome Reception is open to all conference attendees. This is an excellent forum to share information on the day's activities. The reception is an informal and highly interactive environment that gives OOPSLA attendees the opportunity to engage with one another in discussions about relevant, ongoing work and critical issues in key areas.

The Posters program begins with a special session at the Welcome Reception. All posters will be on display and the authors will be present to meet with attendees and discuss their work. The reception gives conference attendees the chance to learn about work in many areas and about preliminary research results. Come meet your old friends and make some new ones.

WikiSym Reception (light refreshments) **Monday, 17:30-19:30** ***Poolside Area near Trellises Rest.***

Mingle with your peers and other folks and talk about whats happening and what to do next.

Newcomer Orientation **Monday, 19:30-20:00** ***Sunrise Room***

There's so much to do and see at OOPSLA that planning a schedule can be daunting, particularly for newcomers. So whether this is your first OOPSLA or you're just feeling overwhelmed, be sure to join us Monday evening for the Newcomer Orientation right after the Welcome Reception. We'll answer your questions and give you tips on how to navigate the conference.

Munch, Muse, Mooch, Mutter, Mix and Mingle at the Mall **Tuesday, 18:00-19:30** ***Fashion Valley Mall***

George Platts Typical meals at the food court range from $5.00 to $10.00.

Computer scientists sometimes find the art of conversation difficult. They tend to spend most of their time with other computer scientists talking about, you guessed it, computer science. I aim to facilitate a number of activities to help people mix and circulate, meet old friends and new people, talk about a variety of subjects. This will be realised by using a "Menu" of instructions and suggestions that all participants will have a copy of, maybe handed to them as they arrive at the Mall. There will be short periods where everybody is visibly and markedly standing on their own silently thinking. There will be periods where people are gathered in pairs, then trios, then quartets, and so on. At times people will be asked to seek out someone that they know, then someone that they have never spoken to before, then form a quartet of known and unknown people. At times it will be very quiet, maybe even silence will reign. At times it will be very briefly very noisy (a "group sneeze" - a synchronised shout of Hishee or Hashee or Hoshee or Hushee sounds like "Atishoo" / a 30" description of a Visit to the Zoo in your mother tongue with gesticulation). Most of the time there will just be the sounds of animated conversations. The event will have a shape (sometimes still, sometimes moving, sometimes huddled, sometimes spaced out), a sound (sometimes silent, sometimes "musical", sometimes a dynamic of whispering then gradually going up to shouting and then gradually going back down to whispering), color (a variety of colored cards will be held up by participants to aid co-ordination). The aim is to create an unusual, stimulating and enjoyable friendly ambience.

A 20th OOPSLA Retrospective **Tuesday, 20:00-21:00** ***Town & Country Room***

This year, OOPSLA is convening for the twentieth time. Join us Tuesday evening as we observe this auspicious occasion with a toast and an elegiac, multimedia look back, featuring a mix of grizzled veterans, object-oriented superstars, and well, anyone with a tale to tell. We'll also be marking another noteworthy milestone, the tenth anniversary of the Java programming language, with a Surprise Guest.

Join us as well at The Wayback Machine, in the courtyard, where an eclectic collection of OOPSLA memorabilia will be on show, and vintage OOPSLA videos will be on display.

A Visit to the San Diego Zoo (Dinner included.) **Wednesday, 19:00-23:00**

The 100-acre (40-hectare) Zoo is home to 4,000 rare and endangered animals representing more than 800 species and subspecies, and a prominent botanical collection with more than 700,000 exotic plants. Please meet the buses outside the Atlas Foyer (near registration) at 7:00pm. Don't be late!

Ice Cream Social and Invitation to OOPSLA 2006 **Thursday, 17:00-19:00** ***Terrace Pavilion & Poolside***

The Ice Cream Social serves as the kick-off event of next year's OOPSLA where you will be able to pick up your OOPSLA 2006 poster and share your 2005 experiences.

NAME BADGES WILL BE REQUIRED TO ATTEND EVENTS.

Demonstrations Tuesday, 11:00-11:45

Chair: Ken Bauer, ITESM, Campus Guadalajara

OOPSLA demonstrations provide an opportunity for companies and universities to show their latest work to an experienced audience. This can be work in progress, commercial applications, proof of concepts, results of academic research, tools, systems or any topic that has interesting object-oriented aspects. Demonstrations are not focused on selling a product but to highlight, explain and present the technical aspects of it. Demonstrators may actively solicit feedback from the usually very technically savvy audience. In the past, this has made for some very interesting demonstration sessions.

14. Automated Debugging in Eclipse

Tuesday, 11:00-11:45 ***Courtyard (Room A)***

Andreas Zeller, Saarland University

Your program fails. What is the cause of this failure? In this demo, we present two delta debugging plug-ins for the Eclipse environment which isolate failure causes in the program history and in the program's execution.

- DDchange is useful if an old (working) version of the unit is available. By systematic tests, it narrows down the set of code changes until it has isolated the failure-inducing change: "The failure was caused by a change to output.cpp at line 197 on June 30th" (Zeller, ESEC/FSE 1999).
- DDstate requires an passing test run of the unit. By narrowing down the differences between program states, it isolates the variables and statements that cause the failure: "At input.java in line 307, this.buffer became NULL, and therefore, the program crashed." (Cleve and Zeller, ICSE 2005).

Both plug-ins are fully automatic; they kick in as soon as a JUnit test fails and produce a diagnosis within minutes. In the demo, we show a number of (buggy) examples, show that their JUnit test fails, and essentially wait until the plug-in produces the diagnostic. (To keep the audience busy, we enable some visualizations showing what's happening behind the scenes.) We are then happy to take questions and suggestions from the audience.

Both plug-ins will be available at http://www.st.cs.uni-sb.de/eclipse/

2. Using Dependency Models to Manage Software Architecture

Tuesday, 11:00-11:45 ***Courtyard (Room B)***

Neeraj Sangal, Lattix, Inc.
Ev Jordan, Lattix, Inc.
Vineet Sinha, Massachusetts Institute of Technology
Daniel Jackson, Massachusetts Institute of Technology

This demonstration will present a new approach, based on the Dependency Structure Matrix (DSM), which uses inter-module dependencies to specify and manage the architecture of software systems. The system is decomposed into a hierarchy of subsystems with the dependencies between the subsystems presented in the form of an adjacency matrix. This form permits succinct definition of design rules to specify allowable dependencies.

The demonstration will show how the DSM can be adapted to represent the architecture of a software system and how the matrix representation is concise, intuitive and appears to overcomes scaling problems that are commonly associated with directed graph representations.

A tool, ArchMap, will be used to demonstrate this approach by loading actual open source Java applications to create DSMs that can represent systems with thousands of classes. We will show how algorithms can be applied to organize the matrix in a form that reflects the architecture and highlights problematic dependencies.

We will demonstrate how design rules can be used to specify and enforce architectural patterns such as layering and componentization. We will also demonstrate the evolution of architecture by creating dependency models for successive generations of Ant, a popular Java utility. Finally, we will demonstrate the application of this approach to the re-engineering of Haystack, an information retrieval system.

3. Io, a small programming language

Tuesday, 11:00-11:45 ***Courtyard (Room C)***

Steve Dekorte

Io is small, pure object oriented, prototype-based programming language. The ideas in Io are mostly inspired by Smalltalk (all values are objects), Self, NewtonScript and Act1 (prototype-based differential inheritance, actors and futures for concurrency), LISP (code is a runtime inspectable/modifiable tree) and Lua (small, embeddable).

Io offers a more flexible language with more scalable concurrency in a smaller, simpler package than traditional languages and it is well suited for use as both a scripting and embedding within larger projects. Io is implemented in C and it's actor based concurrency model is built on coroutines and asynchronous i/o. It employes an incremental tricolor collector and supports weak links and exceptions. Io has bindings for many multiplatform libraries including OpenGL, FreeType, PortAudio and others. This presention will include an overview of the language and demos of some multi-platform desktop applications written with it.

4. Green: A customizable UML class diagram plug-in for Eclipse

Tuesday, 12:00-12:45 ***Courtyard (Room A)***

Carl Alphonce, University at Buffalo
Blake Martin, University at Buffalo

We demonstrate a simple UML class diagram plug-in for Eclipse called Green. While this tool was developed primarily with students and instructors in mind, its extensible architecture makes it potentially useful to others as well.

Green works by directly modifying the Java Development Tool's (JDT) Abstract Syntax Tree (AST) for a type, making class relationships (which are implicit in the AST) explicit in a separate model layer. Changes to the class diagram are reflected in the code directly, since the underlying AST is modified by Green (i.e. Green supports code generation). Changes to Java code files result in changes to the class diagram, since Green is a listener for changes to the underlying AST (i.e. Green supports reverse engineering of diagrams).

Green is also a flexible tool: each binary class relationship is implemented as a separate plug-in to the basic Green tool framework. The set of relationships supported by Green is therefore easily tailored to meet the needs of the user. More importantly, the semantics of the relationships can be defined to suit the user as well. The tool can therefore grow with a student/developer as their needs change.

The demonstration will show Green's code generation, reverse engineering and diagram drawing capabilities. If time permits, its customizability will also be demonstrated.

5. Generics-Related Refactorings for Java in Eclipse

Tuesday, 12:00-12:45 ***Courtyard (Room B)***

Adam Kiezun, MIT
Frank Tip, IBM T.J. Watson Research Center
Robert Fuhrer, IBM T.J. Watson Research Center
Markus Keller, IBM Research

We propose a demonstration of generics-related refactorings for Java, some of which have been recently added to Eclipse 3.1 and presented at ECOOP'05 as a technical paper by the present authors. These refactorings facilitate migration of pre-generic libraries and applications to take advantage of the improved reuse and clarity provided by parametric polymorphism in OO programs.

The automation of the refactorings requires involved static analyses. It also poses challenging usability issues reflecting the complications that programmers face in performing the transformations manually. During the demonstration, we will give an overview of the technical challenges, and highlight the approach we took in addressing the more difficult ones. Our implementation, which we will discuss, is build on top of Eclipse's refactoring and type constraint infrastructure.

We will present:

- Infer Type Arguments - helps in migrating Java application code to generics by determining what concrete types are used for each actual type parameter in the original program, modifying declarations and allocations sites where necessary, and removing casts that have been rendered redundant. This refactoring is included in Eclipse 3.1.
- Introduce Type Parameter - facilitates converting non-generic Java libraries to generics, conveniently and safely for existing clients. It adds a formal type parameter to a class to be used in place of the presently-declared type of a declaration, thus making the class generic. Other necessary changes are performed, possibly including adding new type parameters to related classes. The refactoring infers wildcards which increases the flexibility of the parameterization.

6. MDAbench: A Tool for Customized Benchmark Generation Using MDA

Tuesday, 12:00-12:45 ***Courtyard (Room C)***

Liming Zhu, School of Computer Science and Engineering, University of New South Wales, Australia; Empirical Software Engineering Program, National ICT Australia Ltd.
Ian Gorton, Empirical Software Engineering Program, National ICT Australia Ltd.
Yan Liu, Empirical Software Engineering Program, National ICT Australia Ltd.
Ngoc Bao Bui, Faculty of Information Technology, University of Technology Sydney, Australia

Designing component-based application that meet performance requirements remains a challenging problem, and usually requires a prototype to be constructed to benchmark performance. Building a custom benchmark suite is however costly and tedious due to the complexity and "plumbing" involved in modern component containers and the ad hoc mechanisms they adopt for performance measurement.

This demonstration illustrates an approach for generating customized component-based benchmark applications using a Model Driven Architecture (MDA) approach. All the platform related plumbing and basic performance testing routines are encapsulated in MDA generation "cartridges" along with default implementations of testing logic. We will show how to use a tailored version of the UML 2.0 Testing Profile to model a customized load testing client. The load testing design is logically structured following the testing profile. The performance configuration (such as transaction mix, concurrent users, testing duration and spiking simulations) can also be modeled using the UML model and consequently be generated into code and configuration files. Executing the generated deployable code will collect the performance testing data automatically.

The tool implementation is based on a widely used open source MDA framework AndroMDA. We extended AndroMDA by providing a cartridge for a performance testing tailored version of the UML 2.0 Testing Profile. You can use it to model and generate a load testing suite and performance measurement infrastructure. Essentially, we use OO-based meta-modeling in designing and implementing a lightweight performance testing domain specific language with supporting infrastructure on top of the existing UML testing standard.

7. Testing Domain-Specific Languages in Eclipse

Tuesday, 15:00-15:45 ***Courtyard (Room A)***

Hui Wu, The University of Alabama at Birmingham **Jeff Gray,** The University of Alabama at Birmingham

Domain-specific languages (DSLs) use idioms that are closer to the abstractions found in a specific problem domain. Tool support for testing and debugging DSLs is lacking when compared to the capabilities provided for standard general purpose languages (GPLs). In fact, support for debugging and testing a program written in a DSL is often nonexistent. A common approach for implementing DSLs is to create a pre-processor that translates the DSL source into an object-oriented GPL, such as Java or C++. A DSL grammar serves as the primary artifact for defining DSLs from a higher level of abstraction. This demonstration is focused on a grammar-driven technique to build a testing tool from existing DSL grammars. The DSL grammars are used to generate the hooks needed to interface with a supporting infrastructure written for Eclipse that assists in testing and debugging a program written in a DSL.

This demonstration will introduce a framework to automate the generation of DSL testing tools (e.g., debugger and unit test engine) for imperative and declarative DSLs. Our framework provides Eclipse plug-ins for defining DSLs, along with a translator and interface generator that maps the DSL debugging/testing perspective to the underlying GPL debugging/testing services. This demonstration will provide evidence to support the feasibility and applicability of using the information derived from DSL grammars and existing software components to support end-user debugging and testing in a domain friendly programming environment.

8. XJ : Integrating XML and Java

Tuesday, 15:00-15:45 ***Courtyard (Room B)***

Rajesh Bordawekar, IBM T.J. Watson Research Center **Igor Peshansky,** IBM T.J. Watson Research Center
Michael Burke, IBM T.J. Watson Research Center **Mukund Raghavachari,** IBM T.J. Watson Research Center

XML has emerged as a de facto standard for data integration. Despite the importance of XML, the development of XML processing applications in object-oriented languages such as Java can be tedious and error-prone. Programmers use low-level APIs such as DOM, which provide minimal support for ensuring that programs that process XML are correct with respect to the XML Schemas governing the XML data. Furthermore, the runtime performance of XML processing is a limitation of XML. Runtime libraries do not take advantage of XML Schema information or static analysis of programs to optimize accesses to XML data.

XJ is a research language that integrates XML as a first-class construct into Java. This demonstration will consider the development of an XML processing application in an Eclipse-based environment. It will show how the type system of XML Schema and the expression syntax of XPath is integrated into Java in a manner that is intuitive to both Java and XML programmers, and how XJ supports the development of more robust and efficient XML processing applications. We have built a prototype compiler and a runtime system for XJ (available for download from http://www.alphaworks.ibm.com/tech/xj).

Salient features of XJ include:

1. XML Schemas declarations can be imported in a manner similar to packages and classes in Java.
2. XPath expressions may be written inline. The compiler checks expressions for correctness with respect to XML Schema types.
3. Programmers may embed literal XML to construct XML data.
4. Compiler optimizations to improve the performance of XML processing.
5. Conformance to XML Schema and XPath 1.0 standards

9. An Interactive Visualization of Refactorings Retrieved From Software Archives

Tuesday, 15:00-15:45 ***Courtyard (Room C)***

Peter Weissgerber, Catholic University Eichstaett **Stephan Diehl,** Catholic University Eichstaett

We perform knowledge discovery in software archives in order to detect refactorings on the level of classes and methods. Our REFVIS prototype finds these refactorings in CVS repositories and relates them to transactions and configurations. Additionally, REFVIS relates movements of methods to the class inheritance hierarchy of the analyzed project. REFVIS creates visualizations that show these refactorings in two different layouts:

- the package layout groups the refactored classes according to the packages.
- the class hierarchy layout shows the refactorings in the context of the project's class hierarchy.

We use color coding to distinguish between different different kinds of refactorings. Moreover, our visualizations are interactive as

- they can be zoomed, scrolled and filtered,
- mouse-over-tooltips allow to examine details of the particular refactoring on demand.

10. Eclipse Web and J2EE Development

Tuesday, 16:00-16:45 ***Courtyard (Room A)***

Lawrence Mandel, IBM

Transformed by the Eclipse Web Tools Platform (WTP) project, Eclipse has been extended beyond its Java roots and is now an open source Web and J2EE development IDE. The WTP adds industrial quality Web and J2EE development tools to the Eclipse platform and provides an extensible framework which can be used as the basis for advanced tooling.

From a tools perspective, Web and J2EE objects are contributed as first class citizens in the Eclipse workbench. These objects are used for the creation of Web and J2EE applications using data, server, XML, EJB, J2EE, and Web services tools. From a platform perspective, the WTP includes frameworks that simplify the process of creating new additions to the Eclipse platform such as new servers, XML languages, and Web services implementations. The WTP platform is also customizable using extensions such as the URI resolution framework.

The WTP is an open source, collaborative effort between J2EE companies such as IBM, BEA, ObjectWeb, eteration a.s., and JBoss, and other open organizations such as the Apache Software Foundation, the World Wide Web Consortium (W3C), and the Web Services Interoperability organization (WS-I).

In this demonstration you will see the Eclipse Web tools in action and learn about some of the more useful extensions that the platform provides.

11. PatternStudio - A New Tool for Design Pattern Management

Tuesday, 16:00-16:45 ***Courtyard (Room B)***

Somsak Phattarasukol, California State Polytechnic University, Pomona

Daisy Sang, California State Polytechnic University, Pomona

While design patterns have been widely accepted as best practices in design for a decade, their use is still limited owing to lack of a flexible supporting tool. Many tools are built; however, they have technical restrictions that limit their use at large. For example, they require developers to use a built-in programming language, which may not suit the developers' skills and preferences. They use non-standard serialization techniques; as a result, developers cannot exchange patterns freely across different platforms. Lastly, they are not designed to be easily extendable; developers cannot make any enhancement to match their needs.

We have developed a tool that supports design patterns, called PatternStudio, as proposed in our OOPSLA'04 poster. The tool is an integrated environment specially designed to be language-neutral (multiple code templates can be supplied to generate code in different languages). It supports pattern interchange (patterns are serialized in the XML Metadata Interchange format) and it is extendable (the tool itself is an Eclipse plug-in and can be further extended). Moreover, the tool implements the concept of design verification; it checks whether a derivative model conforms to its originated pattern.

In this demonstration, we will describe how the tool is created from open technologies e.g. Eclipse Modeling Framework (EMF), Graphical Editing Framework (GEF), Java Emitter Templates (JET), and XML Schema (XSD) and show how to work with patterns e.g. define a new pattern, verify a model against a pattern, and generate pattern code in different languages. Also, we will explain how developers can make enhancement by extending the tool.

12. GOOAL: An Educational Graphic Object Oriented Analysis Laboratory

Tuesday, 16:00-16:45 ***Courtyard (Room C)***

Hector Perez-Gonzalez, Universidad Autónoma de San Luis Potosí

Alberto Nunez-Varela, Universidad Autónoma de San Luis Potosí

Jugal Kalita, University of Colorado at Colorado Springs

Richard Wiener, University of Colorado at Colorado Springs

Our goal is to enable rapid production of static and dynamic object models from natural language description of problems. Rapid modeling is achieved through automation of analysis tasks. This automation captures the cognitive schemes analysts use to build their models of the world through the use of a precise methodology. The methodology is based on the use of proposed technique called role posets. and a semi-natural language (called 4W).

First versions of this tool were used as prototypes to produce early design artifacts for very small (toy) problems. Current version has been succesfully used as an educational tool in object oriented software engineering courses.

Original problem statements are automatically translated to 4W language. The produced sentences then, are analyzed with role posets to produce static model views. Finally the 4W sentences are used to generate dynamic views of the problem. This set of methods maximizes analysis process agility, promotes reusability and constitutes a valuable tool in the learning process of object thinking. The prototype tool: GOOAL (Graphic Object Oriented Analysis Laboratory) receives a natural language (NL) description of a problem and produces the object models taking decisions sentence by sentence. The user realizes the consequences of the analysis of every sentence in real time. Unique features of this tool are the underlying methodology and the production of dynamic object models.

A new beta version using problem statements written in spanish will also be presented.

1. CanonSketch and TaskSketch: Innovative Modeling Tools for Usage-Centered Software Design

Wednesday, 11:00-11:45 ***Courtyard (Room A)***

Larry Constantine, Constantine & Lockwood Ltd **Pedro Campos,** University of Madeira

Two closely related novel tools support model-based design and development by addressing major areas of weakness in current modeling tools. CanonSketch is the first tool to support abstract user interface prototyping using canonical abstract components. Canonical abstract prototypes serve as an intermediary between task and object models on the one hand and working user interface prototypes on the other. They have proved to be effective in promoting innovative visual and interaction designs that better support user performance. The tool enables rapid modeling and prototyping through three synchronized views at different levels of abstraction: UML class model, canonical abstract prototype, and functioning HTML prototype.

TaskSketch is an interactive requirements elicitation and modeling tool focused on linking and tracing use cases. It supports collaborative modeling by multiple stakeholders, including clients, marketing staff, and software engineers. It is unique in facilitating the development and exploration of the conceptual architecture based on use case narratives developed in essential form. It enables tracing the requirements of a system, in terms of user intentions and system responsibilities, to the conceptual architecture of that same system, making it easy to extract that architecture from task flows and to prioritize development of the most important classes.

These prototype tools were developed in Objective-C for Mac OS X using object-oriented software engineering techniques such as the Model-View-Controller pattern. Both tools employ industry standard object modeling notation (UML) and compatible extensions and integrate with standard tool suites through XMI export.

KEYWORDS: abstract prototypes, essential use cases, collaborative modeling, model-based design, usage-centered design, UML, XMI

13. AutAT — An Eclipse Plugin for Automatic Acceptance Testing of Web Applications

Wednesday, 11:00-11:45 ***Courtyard (Room B)***

Christian Schwarz, Bekk Consulting AS and IDI/NTNU **Trond Marius Øvstetun,** Mesan AS and IDI/NTNU
Stein Kåre Skytteren, Steria AS and IDI/NTNU

XP (Extreme Programming) practice states that acceptance testing should be automated so they can be run often and facilitate regression testing. The practice also states that it is the customer who should specify these acceptance tests and keep them updated as requirements change. In order to do this, the customer needs a tool that is user-friendly and expressive. Existing frameworks and tools such as FIT and FitNesse support this process, but are based on a textual format for specifying tests and require the customer to write assertions manually.

While FIT and FitNesse are application agnostic, we focus on web-applications. In this demonstration we will present AutAT, an a rich graphical editor on top of FIT, making it more intuitive to specify acceptance tests for automated testing of web-applications. AutAT is an open source Eclipse plugin written using the Eclipse Graphical Editing Framework (GEF). AutAT uses the standard Eclipse plugin development and GEF patterns. Standard FIT tables are generated based on the visual representation of the tests.

AutAT is the result of a master thesis at the Department of Computer and Information Science (IDI) at the Norwegian University of Science and Technology (NTNU) in cooperation with Bekk Consulting AS, Norway. The results of an initial user study AutAT has undergone shows that it is significantly more user-friendly, faster to use and less error prone than using FitNesse + jWebUnit.

The audience will see AutAT as an own perspective in Eclipse, fully functional, though probably without support for all HTML elements. The AutAT GUI is not unlike a standard UML editor.

15. eROSE: Guiding programmers in Eclipse

Wednesday, 11:00-11:45 ***Courtyard (Room C)***

Thomas Zimmermann, Saarland University

Suppose you apply a change to a large software system. What else do you have to change? Missing a required update results in bugs that could easily be avoided.

Just like Amazon.com suggests related products after a purchase, our eROSE plugin for Eclipse guides programmers based on the system history. Suppose you changed an array fKeys[]. eROSE then suggests to change the initDefaults() function - because in the past, both items always have been changed together. If the programmer misses to commit a related change, eROSE issues a warning.

All eROSE needs is a CVS repository; we designed eROSE to be as efficient and unobtrusive as possible. In order to create precise recommendations, eROSE applies efficient data mining techniques: In the Eclipse project with more than 2.9 million lines of code and 300,000 changes, eROSE suggests related changes in less than a second (Zimmermann et al., ICSE 2004). The benefit of eROSE is that it points out item coupling that is undetectable by program analysis - such as between the fKeys[] array and a properties file, or between an SQL query and an image file that contains the database schema.

In the demonstration, we give a brief summary of how eROSE works and then proceed to the actual program - that is, how eROSE is set up (a matter of seconds), and how eROSE makes recommendations. We will focus especially on recommendations that would have been missed by program analysis, and will be happy to take suggestions from the audience.

eROSE is available for download from: http://www.st.cs.uni-sb.de/softevo/erose/

16. Using Globus Grid Objects to Extend a Corba-based Object-Oriented System

Wednesday, 12:00-12:45 ***Courtyard (Room A)***

Scott Spetka, ITT Industries and SUNY Institute of Technology

Richard Linderman, Air Force Research Laboratory

George Ramseyer, Air Force Research Laboratory

The High-Performance Image Exploitation (HIE) FrameWork is an object-oriented system that is designed for management and processing of large images over geographically distributed high-performance computer (HPC) systems. The FrameWork client object implements a CGI user interface to Web browsers using CGICC objects for the client display. The FrameWork originally used Corba for inter-object communications with HPC-based objects. This demo addresses extending the system by incorporating Globus Grid objects.

We added Globus Grid objects to the FrameWork to extend HPC access services. The Grid interface has significant features for controlling remote processing which enhance our distributed computing environment. These include: factory, discovery and notification services; a service browsing interface; and lifecycle management. Globus Grid objects extend the FrameWork to provide loosely coupled Globus services, in addition to tightly coupled Corba interfaces, to support flexible access to HPC systems.

Object-oriented technologies, particularly inheritance, formed the foundation to implement a polymorphic net-centric approach to assure interoperability of our original Corba objects and Globus technologies-based objects and to accommodate future extensions.

The demo will show how we use Corba and Globus objects to distribute HPC computations across multiple heterogeneous HPC systems using innovative scheduling algorithms. Object extensions, implemented using inheritance, allow our architecture to accommodate both technologies and also extend the system to support other distributed computing approaches, like RMI.

This demo will show that our object-oriented architecture is the key to create a hybrid system that presents unique opportunities for parallel system developers to optimize performance for their codes.

Keywords: Grid, Services, Corba, CGI, CGICC, Web, Distributed, Framework High-Performance, Image, Inheritance, Polymorphic

18. SelfSync: A Dynamic Round-Trip Environment Engineering

Wednesday, 12:00-12:45 ***Courtyard (Room C)***

Ellen Van Paesschen, Vrije Universiteit Brussel

Maja D'Hondt, Universite des Sciences et Technologies de Lille

Wolfgang De Meuter, Universite des Sciences et Technologies de Lille

SelfSync is a Round-Trip Engineering (RTE) environment built on top of the object-oriented prototype-based language Self, that integrates a graphical drawing editor for EER diagrams. SelfSync realizes co-evolution between 'entities' in an EER diagram and Self 'implementation objects'. This is achieved by adding an extra EER 'view' to the default view on implementation objects in the model-view-controller architecture of Self's user interface. Both views are connected and synchronized onto the level of attributes and operations.

Development in SelfSync constitutes two phases, both demonstrated via the creation of a small information system. In the first active modeling phase a user draws entities in an EER diagram while corresponding Self implementation objects are automatically created. Changes to an EER entity are in fact changes to a view on one implementation object and thus automatically propagated to this object via Self's reflection mechanism and vice versa.

The second phase is an interactive prototyping process. This phase allows a user to create 'population objects' from the implementation objects created in the previous phase, and initialize them. These population objects are constrained by the multiplicities and the dependencies in the EER domain model. Manipulating them in illegal ways results in automatic warnings of SelfSync.

Intelligent enforcement of constraints on the population objects in a running application are a unique feature of SelfSync, as current RTE environments do not include these objects (instances in class-based languages) in the RTE process. Moreover, as opposed to the state-of-the-art in RTE engineering, SelfSync implements a live link between entities and implementation objects. Changes are first realized at the intended level and then automatically and directly propagated to the other level.

19. TableCode: Defining, Extending and Deploying Object-Oriented Programs Directly from Databases

Wednesday, 14:00-14:45 ***Courtyard (Room A)***

Salleh Diab, TableCode Software Corp. **Yeh Diab,** TableCode Software Corp.

TableCode is a new way to create software based on a metamodelling approach to define an object-oriented program in databases. TableCode extends the OO paradigm to provide an extended data model for the Object, relating the Object to both its Application and Domain. The TableCode demonstration consists of three parts: First, we will show how an OO program can be defined as metadata in databases, presenting an intuitive, clean and easily extendible framework for the source data. A "modeler" program is used to guide a developer (or a domain expert) to create the tables modeled by four schema. This modeler program also contains an internal programming language used only for the class function members.

Second, with the objects now defined in databases, we will show how the OO paradigm can be extended to include application and domain metadata. Specifically, by extending the metadata "vertically" new concepts in OO will emerge, namely "application inheritance" and "software catalog inheritance," similar to class inheritance. Extending the metadata "horizontally" will add other metadata to the Object's metadata definition to represent the real-world object. This part of the demonstration will also show how generics are applied by passing metadata information from higher levels to lower levels, and how this metadata can be retrieved from inside the member functions. (And also how Aspect's "Obliviousness" and "Quantification" concepts can be applied in TableCode.)

Finally, using a virtual machine, we will show how an end-user can deploy an application directly from the databases. Using the application inheritance concept, the end-user can now inherit, intermingle and work with several applications dynamically at run-time - all on a single "Smart" document. We will show how a Universal Unique ID represents an instance of the Object as well as its Application and Domain.

20. CoJava: A Unified Language for Simulation and Decision Optimization

Wednesday, 14:00-14:45 ***Courtyard (Room B)***

Alexander Brodsky, George Mason University **Hadon Nash,** Google

We have proposed and implemented, and will demonstrate, the language CoJava, which offers both the advantages of simulation-like process modeling, and the capabilities of true decision optimization.

By design, the syntax of CoJava is identical to the programming language Java, extended with special constructs to (1) make a non-deterministic choice of a numeric value, (2) assert a constraint, and (3) designate a program variable as the objective to be optmized. A CoJava program thus defines a set of nondeterministic execution paths, each being a program run with specific selection of values in the choice statements. The semantics of CoJava interprets a program as an optimal nondeterministic execution path, namely, a path that (1) satisfies the range conditions in the {\em choice} statements, (2) satisfies the assert-constraint statements, and (3) produces the optimal value in a designated program variable, among all execution paths that satisfy (1) and (2). Thus, to run a CoJava program amounts to first finding an optimal execution path, and then procedurally executing it.

To find an optimal non-deterministic execution path, we have developed a reduction to a standard constraint optimization formulation. Constraint variables represent values in program variables that can be created at any state of a non-deterministic execution. Constraints encode transitions, triggered by CoJava statements, from one program state to the next.

Based on the reduction, we have developed a CoJava constraint compiler. The compiler operates by first translating the Java program into a similar Java program in which the primitive numeric operators and data types are replaced by symbolic constraint operators and data types. This intermediate java program functions as a constraint generator. This program is compiled and executed to produce a symbolic decision problem. The decision problem is then submitted to an external optimization solver.

Keywords: constraints, optimization, simulation

21. Ambient-Oriented Programming in AmbientTalk

Wednesday, 14:00-14:45 ***Courtyard (Room C)***

Jessie Dedecker, Vrije Universiteit Brussel **Sebastián Gonzáles,** Université catholique de Louvain
Stijn Mostinckx, Vrije Universiteit Brussel **Wolfgang De Meuter,** Vrije Universiteit Brussel
Tom Van Cutsem, Vrije Universiteit Brussel **Theo D'Hondt,** Vrije Universiteit Brussel

Software development for mobile devices (such as smart phones and PDA's) is given a new impetus with the advent of mobile networks. Mobile networks surround a mobile device equipped with wireless technology and are demarcated dynamically as users move about. Mobile networks turn the applications running on mobile devices from mere isolated programs into smart applications that can cooperate with their environment. As such, mobile networks take us one step closer to the world of ubiquitous computing envisioned by Weiser; a world where (wireless) technology is gracefully integrated into the everyday lives of its users. Recently, this vision has been termed Ambient Intelligence (AmI for short) by the European Council's IST Advisory Group.

Mobile networks that surround a device have several properties that distinguish them from other types of networks. The most important ones are that connections are volatile (because the communication range of the wireless technology is limited) and that the network is open (because devices can appear and disappear unheraldedly). This puts extra burden on software developers. One of the main reasons for this is that current-day programming languages lack abstractions that deal with the mobile hardware characteristics. This observation justifies the need for a new Ambient-Oriented Programming paradigm (AmOP for short) that consists of programming languages that explicitly incorporate potential network failures in the very heart of their basic computational steps.

AmbientTalk is a first scion of this AmOP programming language family. The power of AmbientTalk lies in the fact that it is simple and expressive. It offers the programmer features to deal with the conceptual properties of wireless networks without having to deal with their technological characteristics. This demonstration shows the code of a program written in AmbientTalk and its runtime behaviour in a real-life context on Smartphones and Laptops.

22. Incremental Exploratory Visualization of Relationships in Large Codebases for Program Comprehension

Wednesday, 15:00-15:45 ***Courtyard (Room A)***

Vineet Sinha, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)
David Karger, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)
Rob Miller, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)

As software systems grow in size and use more third-party libraries and frameworks, the need for developers to understand unfamiliar large codebases is rapidly increasing. In this demonstration, we present a tool, Relo, which supports users understanding by allowing interactive exploration of code. As the developer explores relationships found in the code, Relo builds and automatically manages a visualization mirroring the developer's mental model, allowing them to group viewed artifacts or use the viewed items to ask the system for further exploration suggestions.

The demonstration consists of a few short walkthroughs using Relo on Java code (in the Eclipse IDE). Each walkthrough demonstrates some of the obstacles users face while trying to understand code, and demonstrates corresponding features. The demonstration will hope to convince attendees that the tool is useful for working with large Java projects.

Relo is built on top of the Eclipse IDE and uses the W3C standard, RDF, as the backend representation - allowing to be easily extensible to inclde domain specific information.

Keywords: Program Comprehension, Software Visualization, Large Software Systems

23. fmp and fmp2rsm: Eclipse Plugins for Modeling Features Using Model Templates

Wednesday, 15:00-15:45 ***Courtyard (Room B)***

Krzysztof Czarnecki, University of Waterloo
Michal Antkiewicz, University of Waterloo

- What problems are addressed?

Software product lines, frameworks, and platforms implement different combinations of features for different applications. For example, an e-commerce platform may support different payment methods (credit card, electronic check, purchase order), checkout mechanisms (registered, quick checkout profile, guest), etc. fmp (http://gp.uwaterloo.ca/fmp) is an Eclipse plugin for feature modeling, which allows creating taxonomies of features including variability properties of features (optional, alternative, repeating, etc.) and constraints among them (e.g., implies and excludes). fmp2rsm (http://gp.uwaterloo.ca/fmp2rsm) is a plugin for creating UML model templates in IBM Rational Software Modeller, where individual model elements can be annotated with presence conditions using features from the feature model. fmp allows the user to create a configuration of features that satisfies the constraints of a feature model, and fmp2rsm will automatically create the corresponding instances of model templates. Model templates can be used for representing generic models, e.g., generic business processes or architectures.

- What will the audience be seeing?

We will demonstrate fmp and fmp2rsm using business and architecture models of an e-commerce platform.

- What is object-oriented about the software?

Both systems support object-oriented modeling of software product lines.

- What is unique about the design or implementation?

fmp2rsm automatically colors model templates to indicate which parts of the models correspond to which features.

- What underlying technologies are used?

fmp uses BDDs to propagate constraints and guide users to create correct configurations of features. fmp2rsm generates annotation UML profiles from feature models and uses model transformations to create template instances.

24. A delta-driven execution platform for semantic computing

Wednesday, 15:00-15:45 ***Courtyard (Room C)***

Roly Perera, Dynamic Aspects
Jeff Foster, Dynamic Aspects

We demonstrate the execution model of a computing platform where computation is both incremental and data-driven. We call such an approach delta-driven. The platform is intended as a delivery vehicle for semantically integrated software, and thus lends itself to the semantic web, domain-driven development, and next-generation software development environments. Execution is transparent, versioned, and persistent. This technology - still at an early stage - is called domain/object.

2. Using Dependency Models to Manage Software Architecture

Thursday, 11:00-11:45 ***Courtyard (Room A)***

Neeraj Sangal, Lattix, Inc.
Ev Jordan, Lattix, Inc.
Vineet Sinha, Massachusetts Institute of Technology
Daniel Jackson, Massachusetts Institute of Technology

This demonstration will present a new approach, based on the Dependency Structure Matrix (DSM), which uses inter-module dependencies to specify and manage the architecture of software systems. The system is decomposed into a hierarchy of subsystems with the dependencies between the subsystems presented in the form of an adjacency matrix. This form permits succinct definition of design rules to specify allowable dependencies.

The demonstration will show how the DSM can be adapted to represent the architecture of a software system and how the matrix representation is concise, intuitive and appears to overcomes scaling problems that are commonly associated with directed graph representations.

A tool, ArchMap, will be used to demonstrate this approach by loading actual open source Java applications to create DSMs that can represent systems with thousands of classes. We will show how algorithms can be applied to organize the matrix in a form that reflects the architecture and highlights problematic dependencies.

We will demonstrate how design rules can be used to specify and enforce architectural patterns such as layering and componentization. We will also demonstrate the evolution of architecture by creating dependency models for successive generations of Ant, a popular Java utility. Finally, we will demonstrate the application of this approach to the re-engineering of Haystack, an information retrieval system.

10. Eclipse Web and J2EE Development

Thursday, 11:00-11:45 ***Courtyard (Room B)***

Lawrence Mandel, IBM

Transformed by the Eclipse Web Tools Platform (WTP) project, Eclipse has been extended beyond its Java roots and is now an open source Web and J2EE development IDE. The WTP adds industrial quality Web and J2EE development tools to the Eclipse platform and provides an extensible framework which can be used as the basis for advanced tooling.

From a tools perspective, Web and J2EE objects are contributed as first class citizens in the Eclipse workbench. These objects are used for the creation of Web and J2EE applications using data, server, XML, EJB, J2EE, and Web services tools. From a platform perspective, the WTP includes frameworks that simplify the process of creating new additions to the Eclipse platform such as new servers, XML languages, and Web services implementations. The WTP platform is also customizable using extensions such as the URI resolution framework.

The WTP is an open source, collaborative effort between J2EE companies such as IBM, BEA, ObjectWeb, eteration a.s., and JBoss, and other open organizations such as the Apache Software Foundation, the World Wide Web Consortium (W3C), and the Web Services Interoperability organization (WS-I).

In this demonstration you will see the Eclipse Web tools in action and learn about some of the more useful extensions that the platform provides.

13. AutAT — An Eclipse Plugin for Automatic Acceptance Testing of Web Applications

Thursday, 11:00-11:45 ***Courtyard (Room C)***

Christian Schwarz, Bekk Consulting AS and IDI/NTNU
Stein Kåre Skytteren, Steria AS and IDI/NTNU
Trond Marius Øvstetun, Mesan AS and IDI/NTNU

XP (Extreme Programming) practice states that acceptance testing should be automated so they can be run often and facilitate regression testing. The practice also states that it is the customer who should specify these acceptance tests and keep them updated as requirements change. In order to do this, the customer needs a tool that is user-friendly and expressive. Existing frameworks and tools such as FIT and FitNesse support this process, but are based on a textual format for specifying tests and require the customer to write assertions manually.

While FIT and FitNesse are application agnostic, we focus on web-applications. In this demonstration we will present AutAT, an a rich graphical editor on top of FIT, making it more intuitive to specify acceptance tests for automated testing of web-applications. AutAT is an open source Eclipse plugin written using the Eclipse Graphical Editing Framework (GEF). AutAT uses the standard Eclipse plugin development and GEF patterns. Standard FIT tables are generated based on the visual representation of the tests.

AutAT is the result of a master thesis at the Department of Computer and Information Science (IDI) at the Norwegian University of Science and Technology (NTNU) in cooperation with Bekk Consulting AS, Norway. The results of an initial user study AutAT has undergone shows that it is significantly more user-friendly, faster to use and less error prone than using FitNesse + jWebUnit.

The audience will see AutAT as an own perspective in Eclipse, fully functional, though probably without support for all HTML elements. The AutAT GUI is not unlike a standard UML editor.

19. TableCode: Defining, Extending and Deploying Object-Oriented Programs Directly from Databases

Thursday, 12:00-12:45 ***Courtyard (Room B)***

Salleh Diab, TableCode Software Corp. **Yeh Diab,** TableCode Software Corp.

TableCode is a new way to create software based on a metamodelling approach to define an object-oriented program in databases. TableCode extends the OO paradigm to provide an extended data model for the Object, relating the Object to both its Application and Domain. The TableCode demonstration consists of three parts: First, we will show how an OO program can be defined as metadata in databases, presenting an intuitive, clean and easily extendible framework for the source data. A "modeler" program is used to guide a developer (or a domain expert) to create the tables modeled by four schema. This modeler program also contains an internal programming language used only for the class function members.

Second, with the objects now defined in databases, we will show how the OO paradigm can be extended to include application and domain metadata. Specifically, by extending the metadata "vertically" new concepts in OO will emerge, namely "application inheritance" and "software catalog inheritance," similar to class inheritance. Extending the metadata "horizontally" will add other metadata to the Object's metadata definition to represent the real-world object. This part of the demonstration will also show how generics are applied by passing metadata information from higher levels to lower levels, and how this metadata can be retrieved from inside the member functions. (And also how Aspect's "Obliviousness" and "Quantification" concepts can be applied in TableCode.)

Finally, using a virtual machine, we will show how an end-user can deploy an application directly from the databases. Using the application inheritance concept, the end-user can now inherit, intermingle and work with several applications dynamically at run-time - all on a single "Smart" document. We will show how a Universal Unique ID represents an instance of the Object as well as its Application and Domain.

22. Incremental Exploratory Visualization of Relationships in Large Codebases for Program Comprehension

Thursday, 12:00-12:45 ***Courtyard (Room C)***

Vineet Sinha, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)

David Karger, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)

Rob Miller, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)

As software systems grow in size and use more third-party libraries and frameworks, the need for developers to understand unfamiliar large codebases is rapidly increasing. In this demonstration, we present a tool, Relo, which supports users understanding by allowing interactive exploration of code. As the developer explores relationships found in the code, Relo builds and automatically manages a visualization mirroring the developer's mental model, allowing them to group viewed artifacts or use the viewed items to ask the system for further exploration suggestions.

The demonstration consists of a few short walkthroughs using Relo on Java code (in the Eclipse IDE). Each walkthrough demonstrates some of the obstacles users face while trying to understand code, and demonstrates corresponding features. The demonstration will hope to convince attendees that the tool is useful for working with large Java projects.

Relo is built on top of the Eclipse IDE and uses the W3C standard, RDF, as the backend representation - allowing to be easily extensible to inclde domain specific information.

Keywords: Program Comprehension, Software Visualization, Large Software Systems

DesignFest®

Chair: Daan Hoogland, Luminis B.V.

Featuring MDAFest and once again XPFest.

DesignFest® is an OOPSLA specific conference feature. It is suitable for school and company environment, but it is integrated in no where else as it is in at OOPSLA. It has documented results sinse 1995 and has a more informal history before that.

DesignFest is a training method, not in the courseware sense, but in the sportlike sense. It is an exercise in design with your peers. What it really is, I found can be said in a number of ways.

Some quotes of former DesignFest chairs:

"DesignFest is a free event (for conference registrants) created to give OOPSLA attendees the opportunity to learn more about design by doing it."

"The OOPSLA DesignFest is about design and creativity."

"DesignFest is not about passively sitting and listening to experts talk about design."

"DesignFest is about sharpening your design skills by rolling up your sleeves and working on a real problem with others in the field."

In a DesignFest groups are formed to work on design problems. A group does not get to see its full problem description until the moment the DesignFest starts. However, the group is free to communicate beforehand to agree on process if they so desire. Each member of the team assumes a specific role. The group is expected to produce a set of deliverables by the end of the session.

Of course roles and deliverables may vary when special processes are agreed upon. This is certainly true for MDAFest and XPFest but is not limited to such specific settings. We will provide a wiki area for each group where you can discuss and document your process and deliverables. If you agree we will also give your email adres to your fellow group members.

Like in the past few years, a limited number of teams will be able to use EXtreme Programming practices in the XPFest. New this year is the addition of an MDAFest, where participants will try their skills using Model Driven Architecture. The MDAFest is not a tutorial on MDA; it is an opportunity for practitioners to discuss practices and tools, working on a specific problem.

In our experience people do not want to start their designfest during the keynote. Therefore this year we will have a 3/4-day session on Wednesday that starts right after the keynote.

Sunday, 08:30-17:00	***San Diego Room***	**Wednesday, 10:30-17:00**	***Sunset***
Sunday, 08:30-12:00	***San Diego Room***	**Wednesday, 13:30-17:00**	***Dover & Stratford rooms***
Sunday, 13:30-17:00	***San Diego Room***		

Problem Descriptions

Teamwork for Physical Therapists of Children

The problem is to design a system that can help a small team of physical therapists that treat children, to communicate between team members, with doctors and with parents.

When treating children, continuity is very important, communication with parents plays a role, and the children themselves are often not able to participate in the planning and keeping track of treatment as an adult would do.

The first and most important requirement of the team of physical therapists is to have a system that would allow them to transfer work of one therapist to another (in case of absence or illness of a therapist) seamlessly to ensure continuity of the therapy. Since this would require already a fair amount of data about treatments for each child to be kept, the team would like to be able to put this data to additional use for communication with doctors and parents, and to keep appointment schedules.

On-Line Image Management

The proposal involves the design of a system that manages digital images as a Web-service, specifically focused for use by larger or international organizations. The system will be used by professional photographers to deliver their photos and by print shops, both possibly external to the organization that deploys the image system.

Many organizations of any size own images to be used for company brochures, advertising material, and internal or external publications. During the last few years these images are increasingly kept on digital media.

The management of these digital images poses problems that are different from managing a set of text-documents, for example:

- Images take much more space than documents, typically 10-40MB.
- Additional information about an image needs to be stored external to the image to allow searching.
- The copyright of an image may reside outside the company that uses it.

New technology and standards make it feasible to provide management of images as a web-service. This enables organizations to have loose relationships with image providers, and media providers to offer flexible contracts, customized towards the need of a specific organization.

Identity Management Reference Architecture

In this design fest we will come up with a design for identity management for a large enterprise. Many large companies have a number of different systems doing identity management in different ways. Some systems use LDAP, some use RDBMS and some others use Active Directory as identity store.

Different systems are currently in production and they use ERP systems, a number of COTS products and Java/J2EE. LDAP and Active Directory are being kept in Sync using synchronization tools. A user that is added to one system may need to be added to multiple systems with different privileges. Another challenge is to provision two million volunteers. Identity products from different vendors are being considered.

Here is the general workflow of the system to be built:

Employees are first provisioned in HR system and user identity flows to Active Directory followed by LDAP. Volunteers are usually provisioned using the provisioning tool which propagates identity with appropriate privileges. User is provided with password synchronization across all the systems. When a user does not need access, they are de-provisioned automatically. Some systems need single sign on capability and this is provided by policy server(s) from vendors.

Provisioning an internet provider's network

This problem has been introduced into the domain of telecom providers and cable-television suppliers as they broadened their range of products and started to delve into each others markets.

The maintenance people of a service provider have to administer and configure devices and daemons running on server machines. This can be quite cumbersome and is a good candidate for automation.

A system for configuring and maintaining a network of devices for routing and processing and receiving ip data must be maintained. A lot of it is automated. But this is not standard of the shelf software. Big and small service providers often build their own.

The systems involved run very different hardware. Some are configurable through snmp interfaces others through html over http interfaces. Again others run commandline interfaces on proprietary operating systems.

Extreme Programming Production of a Tournament Management System

In this project, the DesignFest participants will use the Extreme Programming (XP) methodology to design and build a working tournament management system within one day. The organizers will provide ongoing instruction in the XP practices as appropriate, and will also act as the customer for the software project. The day will proceed as a series of iterations. During each iteration, the customers and designers will "negotiate" the features of the system that are to be developed, coding will take place, and the customer will be given the chance to accept or reject the delivered system. At the end of each iteration, the customer should have a working system that incorporates the features that were previously negotiated.

Manufacturing / Printing Process Simulator

Designers of real-time embedded software are often faced with a serious problem. The physical system that their software will control is being designed concurrently and is therefore not available as a test platform for the software. Add to this the usual dangers of running untested software on systems that have inherent safety and property damage risks, and the arguments in favor of simulation environments become quite strong.

Design Fest teams choosing this assignment will design a general-purpose simulator for a mechanical process. This proposal addresses a specific domain within embedded systems: that where software must control the transport of physical objects through various way points where physical actions are applied to them.

Software is needed to simulate the physical system so that the software being designed to control that system can be empirically tested. The control software is not part of this proposal.

Doctoral Symposium Monday

Chair: Fred Grossman, Pace University

The goal of the OOPSLA 2005 Doctoral Symposium is to provide useful guidance for the completion of the dissertation research and the initiation of a research career. The Symposium will provide an interactive forum for doctoral students in one of two phases in their doctoral progress.

Apprentices

Students who are just beginning their research, are not ready to actually make a research proposal, but are interested in learning about structuring research and getting some research ideas.

Proposers

Students who have progressed far enough in their research to have a structured proposal, but will not be defending their dissertation in the next 12 months. This 12-month stipulation is set in place because we would like for the students to have sufficient time to incorporate the advice and suggestions discussed in the symposium.

This year, the mentors are Fred Grossman (Pace University), Craig Chambers (University of Washington), Ron Frank (Pace University), Robert Kessler (University of Utah), Ole Madsen (Alexandra Instituttet A/S, Aarhus University). To provide further opportunity for discussion and feedback, doctoral symposium presenters have a poster on display during the conference and a two-page short paper published in the Conference Companion proceedings.

Due to the mentoring nature of the event, it is only open to those selected for participation.

Monday, 8:30-17:00 ***Stratford***

Software Architecture Improvement through Test-Driven Development

David Janzen, *University of Kansas*

Component-Based Software Engineering: a Quantitative Approach

Miguel Goulão, *Universidade Nova de Lisboa*

Using Refactorings to Automatically Update Component-Based Applications

Daniel Dig, *UIUC*

Inferring Context-Free Grammars for Domain-Specific Languages

Faizan Javed, *University of Alabama at Birmingham*

High-Level Declarative User Interfaces

Sofie Goderis, *Vrije Universiteit Brussel*

Fortune Teller: Improving Garbage Collection Performance in Server Environment using Live Objects Prediction

Feng Xian, *University of Nebraska-Lincoln*

System Level Perspective on Object Locality

Carl Lebsack, *Iowa State University*

Adaptive Compositions across Organizational Boundaries

Thomas Cottenier, *Illinois Institute of Technology*

Tuesday Dynamic Languages Symposium

Chair: Roel Wuyts, Université Libre de Bruxelles

Java and C#, the latest mainstream static languages, popularized to a certain extent dynamic language features such as garbage collection, portability and (limited forms of) reflection. In the near future, we expect this dynamicity to increase even further. E.g., it is getting clearer year after year that pervasive computing is becoming the rule and that concepts such as meta programming, reflection, mobility, dynamic reconfigurability and distribution are becoming increasingly popular.

The goal of this symposium is to provide a highly visible, international forum for researchers working on dynamic features and languages. Papers will be presented that were selected by a broad international program committee with respected members from the functional, logic and object-oriented language communities. To top it off there will be a keynote speech by Gerald Jay Sussman.

More information is at the Dynamic Languages 2005 web site.

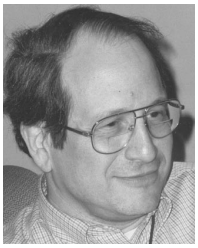
Welcome **Tuesday, 08:30-09:00** ***Sunset***

Objects as Software Services **Tuesday, 09:00-10:00** ***Sunset***

Gilad Bracha

Language Mechanisms and Extensions **Tuesday, 10:30-12:00** ***Sunset***

Why Programming is a Good Medium for Expressing Poorly Understood and Sloppily Formulated Ideas **Tuesday, 13:30-15:00** ***Golden West Room***



Gerald Jay Sussman, Professor, Massachusetts Institute of Technology

I have stolen my title from the title of a paper given by Marvin Minsky in the 1960s, because it most effectively expresses what I will try to convey in this talk.

We have been programming universal computers for about 50 years. Programming provides us with new tools to express ourselves. We now have intellectual tools to describe "how to" as well as "what is". This is a profound transformation: it is a revolution in the way we think and in the way we express what we think.

For example, one often hears a student or teacher complain that the student knows the "theory" of some subject but cannot effectively solve problems. We should not be surprised: the student has no formal way to learn technique. We expect the student to learn to solve problems by an inefficient process: the student watches the teacher solve a few problems, hoping to abstract the general procedures from the teacher's behavior on particular examples. The student is never given any instructions on how to abstract from examples, nor is the student given any language for expressing what has been learned. It is hard to learn what one cannot express. But now we can express it!

Expressing methodology in a computer language forces it to be unambiguous and computationally effective. The task of formulating a method as a computer-executable program and debugging that program is a powerful exercise in the learning process. The programmer expresses his/her poorly understood or sloppily formulated idea in a precise way, so that it becomes clear what is poorly understood or sloppily formulated. Also, once formalized procedurally, a mathematical idea becomes a tool that can be used directly to compute results.

I will defend this viewpoint with examples and demonstrations from electrical engineering and from classical mechanics.

Gerald Jay Sussman is the Matsushita Professor of Electrical Engineering at the Massachusetts Institute of Technology. He received the S.B. and the Ph.D. degrees in mathematics from the Massachusetts Institute of Technology in 1968 and 1973, respectively. He has been involved in artificial intelligence research at M.I.T. since 1964. His research has centered on understanding the problem-solving strategies used by scientists and engineers, with the goals of automating parts of the process and formalizing it to provide more effective methods of science and engineering education. Sussman has also worked in computer languages, in computer architecture and in VLSI design.

Sussman is a fellow of the Institute of Electrical and Electronics Engineers (IEEE). He is a member of the National Academy of Engineering (NAE), a fellow of the American Association for the Advancement of Science (AAAS), a fellow of the American Association for Artificial Intelligence (AAAI), a fellow of the Association for Computing Machinery (ACM), a fellow of the American Academy of Arts and Sciences, and a fellow of the New York Academy of Sciences (NYAS). He is also a bonded locksmith, a life member of the American Watchmakers-Clockmakers Institute (AWI), a member of the Massachusetts Watchmakers-Clockmakers Association, a member of the Amateur Telescope Makers of Boston (ATMOB), and a member of the American Radio Relay League (ARRL).

Common Lisp: a dynamic language for a dynamic world **Tuesday, 15:30-16:30** ***Sunset***

Jans Aasman

Common Lisp: a dynamic language for a dynamic world; you don't have to sacrifice performance for programmer productivity

Implementing Dynamic Languages on the .NET CLR **Tuesday, 15:30-17:00** ***Sunrise***

John Gough

Language Reasoning **Tuesday, 16:30-17:30** ***Sunset***

I Have Nothing to Declare but My Genius **Tuesday, 17:30-18:30** ***Sunset***

Brian Foote

Educators' Symposium — Monday Morning

Chair: Eugene Wallingford, University of Northern Iowa

Join us at OOPSLA's 14th forum for object-oriented educators and trainers! This one-day symposium offers a venue for educators and trainers to share their experiences teaching object-oriented technology and to explore ideas that can help us understand and teach OO technology better.

This year's symposium offers new ideas for addressing the problems faced by OO educators. We will have presentations of teaching ideas, demonstrations of OO teaching tools, a poster session—and plenty of active sessions in which attendees work together to put these ideas into practice.

This year, we are honored to have as our keynote speaker Ward Cunningham. Ward has pioneered many of the ideas and tools we all use today: object-oriented programming, CRC cards, patterns, wiki, extreme programming, test-first design, and FIT. In his talk, Ward will offer some advice for how we can recognize good ideas in their humility, how to nurture good ideas to fruition, and how we might teach these skills in our courses.

OOPSLA is again delighted to announce the availability of scholarships for educators from 2- and 4-year colleges to attend the OOPSLA 2005 and the Educators' Symposium. These scholarships are sponsored by the Association of Computing Machinery (ACM) Special Interest Group on Programming Languages (SIGPLAN). For more information, see the Educator Scholarship page.

Agile Teaching — Monday, 08:30-10:00 — *San Diego Room*

The symposium begins with an interactive presentation that illustrates agility in the classroom via a Scrum sprint with student apprentices.

Opening Remarks

Eugene Wallingford, *University of Northern Iowa*

Apprenticeship Agility in Academia

David West, *New Mexico Highlands University*
Pam Rostal, *New Mexico Highlands University*

Teaching Techniques — Monday, 10:30-12:00 — *San Diego Room*

This session consists of two sets of papers. The first two show ways to support active learning in the classroom, to involve students actively in the learning of OO material. The second two show ways to make object-oriented design concrete, something that students can grasp and practice more immediately. Each pair of presentations will be followed by ample time for the authors and attendees to discuss the ideas under consideration.

A Laboratory for Teaching Object-Oriented Language and Design Concepts with Teachlets

Axel Schmolitzky, *University of Hamburg, Germany*

Teaching OO Methodology in a Project-Driven CS2 Course

Barry Kurtz, *Appalachian State University*
James Wilkes, *Appalachian State University*
Frank Barry, *Appalachian State University*

Modeling OO Design

Robert Rist, *University of Technology, Sydney*

Roles of Variables in Object-Oriented Programming

Petri Gerdt, *University of Joensuu, Department of Computer Science*
Jorma Sajaniemi, *University of Joensuu, Department of Computer Science*
Pauli Byckling, *University of Joensuu, Department of Computer Science*

Keynote Address — Monday, 13:30-15:00 — *San Diego Room*

This year, we are honored to have as our keynote speaker Ward Cunningham. Ward has pioneered so many of the ideas and tools we all use today: object-oriented programming, CRC cards, patterns, wiki, extreme programming, test-first design, and FIT. In his talk, Ward will offer some advice for how we can recognize good ideas in their humility, how to nurture good ideas to fruition, and how we might teach these skills in our courses.

Nurturing the Feeble Simplicity

Ward Cunningham, *Microsoft Corporation*

Teaching Tools **Monday, 15:30-16:15** ***San Diego Room***

This session consists of a paper session and a panel. The papers explore ways that educators are using two of Ward's innovations, CRC cards and FIT tests, in the classroom. The panel looks to the future of OO education, in particular seemingly never-ending tug-of-war around language and programming style in the first-year courses. Both of these subsessions will be contain ample time for the authors, panelists, and attendees to discuss the ideas under consideration.

The Practice of Specifying Requirements Using Executable Acceptance Tests in Computer Science Courses

Grigori Melnik, *University of Calgary* Frank Maurer, *University of Calgary*

Improving CRC-Card Role-Play with Role-Play Diagrams

Jürgen Börstler, *Umeå University*

Looking to the Future **Monday, 16:15-17:30** ***San Diego Room***

The symposium closes with a panel discussion and an open-mike session. The panel looks to the future of OO education, in particular seemingly never-ending tug-of-war around language and programming style in the first-year courses. The open-mike session session gives participants a chance to share their best teaching ideas and their ideas for future Educators's Symposia.

Are We Doomed? Reframing the Discussion

TBA

Closing Session

Eugene Wallingford, *University of Northern Iowa.*

Demos and Posters **Monday, 8:30-17:30** ***San Diego Room***

During coffee breaks and the long lunch period, symposium attendees have the opportunity to see demonstrations of tools that can help us teach OOP better and to interact with presenters of ideas that can change how we approach our courses.

Virtual PC: Letting Your Students Install and Explore Without Fear

Joe Hummel, *Lake Forest College, Dept of Math/CS*

A Pattern-based Approach for Introducing Object-Oriented Analysis and Design in Undergraduate Courses

Haitham Hamza, *University of Nebraska-Lincoln*

Green: A Pedagogically Customizable Round-tripping UML Class Diagram Eclipse Plug-in

Carl Alphonce, *University at Buffalo, State University of New York* Blake Martin, *University at Buffalo, State University of New York*

Visual OS: An Object-Oriented Approach to Teaching Operating System Concepts

James Hill, *Institute for Software Integrated Systems, Vanderbilt University* Aniruddha Gokhale, *Institute for Software Integrated Systems, Vanderbilt University*

Teaching Web Services with Water

Matt Kendall, *North Carolina State University* Ed Gehringer, *North Carolina State University*

Double Dispatch for Two-Tiered Type Checking

Adair Dingle, *Seattle University*

KlassroomSwing: A GUI Package for Students

Dean Sanders, *Northwest Missouri State University*

Implementing Operating Systems the Object-Oriented Way

Juan Carlos Guzman, *Department of Computer Science, Southern Polytechnic State University* Patrick Bobbie, *Department of Computer Science, Southern Polytechnic State University*

Essays

Chair: Brian Marick, Exampler

From the moment the last four letters of its name were chosen, OOPSLA has been a place where members of diverse communities bring their special obsessions, toss them into a rich nutrient bath of attendees, and watch what grows. The Essays track, new for 2005, aims to enhance that ongoing experiment by using the special characteristics of the essay form. An essay demonstrates or describes the way an interesting mind grapples with an idea. The mind and the process of grappling are as important as the idea. Our ideal essay displays an unfamiliar way of thinking being applied to a topic that deserves the attention of the OOPSLA community.

At the conference, essay presentations will prompt listeners to approach the rest of the conference in a slightly different way, wondering, "How would a person who thought—that—way approach the issue we're talking about?" To that end, each essayist's presentation will be followed by a discussant's invited reaction to the essay, and then discussion with the audience.

A Simple Model of Agile Software Processes - or - Extreme Programming Annealed — **Tuesday, 10:30-12:00** — ***Sunrise***

Glenn Vanderburg, Countrywide Financial

The essay begins:

> *"If you built a piece of software that was as tightly coupled as Extreme Programming, you'd be fired." It was late 1999, and I was sitting at lunch with Pragmatic Dave Thomas and the rest of the North Texas XP interest group. It's not unusual for Dave to make provocative statements like that, but this time I was dumbstruck [...] He was right. The same characteristic that I appreciated in XP, I would decry in a software design.*

In his essay, Glenn tackles various questions: Are the interdependencies between Extreme Programming's various practices bad? If not, what makes coupling in a process different from coupling in a product? And can understanding the nature of XP's coupling help us tailor software processes?

FlashBoFs

Chair: Daan Hoogland, Luminis B.V.

Do you want to do something and need brothers in arms? Propose a BoF. BoF (Birds of a Feather) session (for those that do not know) are highly unorganised session of people with a common goal. This can be exploring, defining a standard or programming some code, to name just a few.

FlashBoFs where first introduced at the 2004 OOPSLA (2004 Flash BOF home page) and received well.

Instead of using written announcements, Monday, Tuesday and Wednesday evenings of the conference FlashBoFs will be held. Each FlashBoF session will last about 90 minutes. Several rooms have been set aside each evening for FlashBofs—people are invited to announce their BoFs with time and place at the conference. A conference Wiki is made available for the BoF organiser to display their BoF results.

At the FlashBoF sessions you can call upon your peer to join you in your session. If you are not yet sure if you know what it is exactly come to a flashbof session and try to find your peer interactively. After finding each other and your common ground you can break-out and start your own session.

BoFs	**Monday, 17:00-22:00**	***several locations***
FlashBoFs	**Tuesday, 17:00-18:30**	***Royal Palm Salon 1 and 2***
FlashBoFs	**Wednesday, 17:00-18:30**	***Sunset***

Instant Arts School Experience - just add people

George Platts

The “Instant Arts School Experience” will offer delegates both a range of stimulating input from across the arts and an opportunity for interactive and experiential creation.

George Platts will be showing excerpts from a varied collection of films alongside (mainly) silent artist’s films / video work, accompanied by a selection of unusual musics. There may also be an opportunity for delegates to make films and installations with equipment and art materials that will be available. What follows is a partial list of films and music that will be on hand.

Tuesday, 10:30-12:00	***Courtyard***
Tuesday, 15:30-17:00	***Courtyard***
Wednesday, 10:30-12:00	***Courtyard***
Wednesday, 13:30-15:00	***Courtyard***
Thursday, 10:30-12:00	***Courtyard***
Thursday, 13:30-15:00	***Courtyard***

Films and video works : -

Gerry : Gus Van Sant (USA)
Atanarjuat - The Fast Runner : Zach Kunuk (Canada)
The Straight Story : David Lynch (USA)
Rififi : Jules Dassin (France)
Films by Aki Kaurismaki (Finland)
Der Lauf der Dinge : Peter Fischli / David Weiss (Switzerland)
Decasia : Bill Morrison (USA)
Tango : Zbigniew Rybczynski (Poland)
Downside Up and other experimental films : Tony Hill (UK)
Moving Photographs, Long Exposures : George Platts (UK)
-various videoworks - : Bill Viola (USA)

Music :-

Rhys Chatham (USA)
Low (USA)
The Necks (Australia)
John Cage (USA)
Robert Ashley (USA)
Philip Glass (USA)
Steve Reich (USA)
Simeon ten Holt (The Netherlands)
Oum Kalsoum (Egypt)

Lightning Talks

Thursday, 13:30-15:00	**Sunrise**

Co-Chair: Bjorn Freeman-Benson, Eclipse Foundation

Co-Chair: Bruce Horn, Ingenuity Software

A Lightning Talk is a 5-minute presentation on any topic of interest to the OOPSLA community. Lightning Talks will be presented back to back with strict enforcement of the 5-minute time limit and a few acetate foils (or their electronic equivalent).

Co-chair: Elisa L. A. Baniassad, Chinese University of Hong Kong

Co-chair: James Noble, Victoria University of Wellington

The Onward! track provides a forum for visions of the future of our field. Onward! is the place to reveal the revolutionary, air the provocative, and expose the subversive. The Onward! track is composed of three sub-tracks: papers, presentations, and films.

Onward! papers are included in the technical proceedings, and are part of the main research program. Of the 27 papers submitted to Onward!, two were selected for inclusion in the papers track.

Onward! presentation can describe: new paradigms or metaphors in computing, new thinking about objects, new framings of computational problems or systems, intriguing technologies, emerging methodologies, and late-breaking research. Onward! presentations are provocative, inspiring, interesting, amusing, irritating, and in some cases subversive. Presentations are submitted as 10-page research papers, and appear in the conference companion. Presentations provide a venue for those exciting and groundbreaking ideas that are not yet ready for inclusion in a regular technical track.

Onward! films investigate ideas, concepts, insights, or almost anything related to programming. They can take almost any form: documentary on learning a new language; interviews with practitioners; animation/depiction of your new technique; diary of a frustrated grad student.

Onward! Papers

Tuesday, 10:30-12:00 ***Town & Country Room***

Subtext: Uncovering the Simplicity of Programming

Jonathan Edwards, MIT

Representing programs as text strings makes programming harder then it has to be. The source text of a program is far removed from its behavior. Bridging this conceptual gulf is what makes programming so inhumanly difficult—we are not compilers. Subtext is a new medium in which the representation of a program is the same thing as its execution. Like a spreadsheet, a program is visible and alive, constantly executing even as it is edited. Program edits are coherent semantic transformations.

The essence of this new medium is copying. Programs are constructed by copying and executed by copy flow: the projection of changes through copies. The simple idea of copying develops into a rich theory of higher-order continual copying of trees. Notably absent are symbolic names, the workhorse of textual notation, replaced by immediately-bound explicit relationships. Subtext unifies traditionally distinct programming tools and concepts, and enables some novel ones. Ancestral structuresare a new primitive data type that combines the features of lists and records, along with unproblematic multiple inheritance. Adaptive conditionalsuse first-class program edits to dynamically adapt behavior.

A prototype implementation shows promise, but calls for much further research. Subtext suggests that we can make programming radically easier, if we are willing to be radical.

X10: An Object-Oriented Approach to Non-Uniform Cluster Computing

Philippe Charles, IBM Research
Christopher Donawa, IBM Software Group
Kemal Ebcioglu, IBM Research
Christian Grothoff, Purdue University
Allan Kielstra, IBM Software Group
Vijay Saraswat, IBM Research
Vivek Sarkar, IBM Research
Christoph von Praun, IBM Research

The next generation of high performance computers (e.g. those capable of $O(10^{15})$ operations per second) will be based on scale-out techniques rather than increasing clock rates. This leads to a notion of clustered computing: a single computer may contain hundreds of thousands of tightly coupled nodes. Unlike a distributed model, failure of a single node is tantamount to failure of the entire machine. However, the cost of memory access by a hardware processor may vary as much as five orders of magnitude across the cluster; hence the notion of a single shared memory may no longer be appropriate for such machines.

We have designed a concrete modern object-oriented programming language, **X10**, for high performance, high productivity programming of such machines. Past work in the Java Grande Forum has exposed the need for substantial changes in modern OO languages in order to support high performance computation (e.g. support for true multi-dimensional arrays, value types, relaxed exception model, changes to the concurrency model, support for distribution etc.) X10 builds on this past work. A member of the partitioned global addressspace-family of languages (which includes **Titanium**, **UPC**, and **Co-ArrayFortran**), **X10** is distinguished by the explicit reification of locality (places), termination detection (finish), by the use of lock-free synchronization (conditional atomic sections), and by the ability to express clustered data-structures (e.g. arrays scattered across multiple places). **X10** smoothly integrates concurrent shared memory access (e.g. as expressed by OpenMP or Java threads) with message-passing (e.g. as expressed by MPI). We present the design of the core features of the language, experience with a reference implementation and present results from some initial productivity studies.

Onward Presentations 1: Exploration

Tuesday, 13:30-15:00 *Town & Country Room*

Constructing a Metacircular Virtual Machine in an Exploratory Programming Environment

David Ungar, Sun Microsystems Laboratories **Alex Ausch,** Sun Microsystems Laboratories
Adam Spitz, Sun Microsystems Laboratories

Can virtual machine developers benefit from religiously observing the principles more often embraced for exploratory programming? To find out, we are concurrently constructing two artifacts "a Self VM entirely in Self (the Klein VM), and a specialized development environment" with strict adherence to pure object-orientation, metacircularity, heavy code reuse, reactivity, and mirror-based reflection. Although neither artifact is yet complete, the environment supports many remote debugging and incremental update operations, and the exported virtual machine has successfully run the whole compiler.

As a result of our adherence to these principles, there have been both positive and negative consequences. We have been able to find and exploit many opportunities for parsimony. For example, the very same code creates objects in the bootstrap image, builds objects in the running VM, and implements a remote debugger. On the other hand, we have been forced to expend effort to optimize the performance of the environment. Overall, this approach trades off the performance of the environment against the architectural simplicity and ease of development of the resulting VM artifact. As computers continue to improve in performance, we believe that this approach will increase in value.

Exploring the Acceptability Envelope

Martin Rinard, MIT **Cristian Cadar,** Stanford **Huu Hai Nguyen,** MIT/SMA

An acceptability envelope is a region of imperfect but acceptable software systems surrounding a given perfect system. Explicitly targeting the acceptability envelope (rather than attempting to minimize the number of errors, as is the current practice) has several potential benefits. Specifically, leaving acceptable errors in the system eliminates the risks and costs associated with attempting to repair the errors; investing fewer resources in less critical regions of the program and more resources in more critical regions may increase acceptability and reduce the overall investment of development resources.

To realize these benefits, the acceptability envelope must be both sizable and accessible. We present several case studies that explore the acceptability envelopes of the Pine email client and the SurePlayer MPEG decoder. These studies show that both Pine and SurePlayer can tolerate the addition of many off-by-one errors without producing unacceptable behavior. This result suggests that current systems may be overengineered in the sense that they can tolerate many more errors than they currently contain.

Our SurePlayer case study also shows that SurePlayer has unforgiving regions of the code that must be close to perfect for the system to function at all. To effectively exploit the acceptability envelope, developers must be able to distinguish forgiving and unforgiving regions so that they can appropriately prioritize their development effort. In SurePlayer, the unforgiving regions tend to occur in code that uses metadata to parse the input stream; the forgiving regions tend to access the data within each image. This result suggests that developers may be able to use relatively simple indicators to effectively prioritize their development effort.

Breakthrough Ideas

Tuesday, 15:30-17:00 *Town & Country Room*

Jim Coplien **Andrew Hunt** **Bonnie Nardi** **Glenn Vanderburg**
Brian Foote **Crista Lopes** **Rob Tow** **Dave Thomas**
Richard Gabriel **Brian Marick** **Dave Ungar**

A Breakthrough Idea is a short essay/talk designed to stimulate thought and reflection among the computing community. The authors above were asked to respond to a set of questions posed by the co-chairs of Onward!. The authors could respond to any one of the questions, any set of the questions, or any other thought that came to mind after reading the questions. The authors were encouraged to use whatever genre or format they chose. The following were the questions:

- What do you hope is false, but fear is true?
- What do you believe shouldn't be invented?
- If you could delete any idea, technology, or event from computing history, what would it be?
- If you could inject any idea, technology, or event into computing history, what would it be? What historical figure do you wish were here today and working on software?
- What established principle in programming is overused and why do you think so?
- What one is underused and why do you think so?
- What speech would you want to make to all the software people gathered at the base of the Himalayas?
- What speech would you want to make to all the world leaders there?
  - ...to all the CEOs?
  - ...to all the professors?
  - ...students?
- What is the least understood thing in computing that if only it were understood, everything would change?

Comfortable as Blue Jeans

James O. Coplien, DAFCA

It fits
Part of me, yet not me.
Unplugged: oxygen its food
And the breeze its power adaptor
Its identity mine, yet not me
    I put it down,I take it up
    like a well-worn wallet plump
    with life's means and memories
A thin mask worn in life's play
    and both of us are players
I and my computer are one; it does not call me User
a curse unique to programmers and other pushers
It remembers the travails of its forebearers
Laboring under companion's fingers
35 mechanical levers linking brain to digits
slow and tired bones, the machines of the information age...

Picture this: my new machine, a soulmate,
    In conversation unencumbered by flesh and bone
Mind to mind
It slumbers not, nor sleeps
Its software penned by
    My soul
And like my soul,
its hardware Im-mortal, invisible
    Us only wise
Transparent personae mine, yet not me,
That with me loves, and lives, and walks life's journey
In a great and beautiful field
Both real and virtual at once
    as only a True and rich life is
With me it dies
    —our memories live on in our networks
No duality of community here
    and Internet there
        The network is the com-
            munity
An invisible mask the birthright of all humanity
One Web of life

Collected Thoughts

Brian Foote, University of Illinois

In Praise of "Cut 'n' Paste"

"Cut 'n' Paste", the practice of creating a new software component or application by making a source copy of an existing body of code and "beating on it" mercilessly until it meets some new set of requirements, is often treated with disdain and contempt by the traditional software engineering community and (post-)modern object-oriented methodologists alike. Despite this, the ubiquity and enduring popularity of this approach begs the question: What are these people doing right?

Evidence suggests that cut 'n' paste is a nearly essential, and nearly universal, practice when new framework applications are constructed, be they for GUI code, Eclipse plug-ins, or what have you. Adaptations to complex applications such as compilers often demand an incremental, test-as-you-go approach as well.

The advantages to this approach, naturally, are the complements of those associated with factoring out duplication. A copy provides a safe sandbox off to the side of a vital development stream. Changes made to copies cannot disrupt production code, and like duplicate genes, copies can evolve new functionality without jeopardizing existing functionality in the original. Mere copies can avoid, for a time, the intricacies associated with engineering away duplicate functionality.

Indeed, expedient code copying has long been observed to be characteristic of an early, exploratory phase in the evolution of object-oriented components and systems, but its virtual indispensability to framework developers has been less widely acknowledged.

Big Bucket of Glue

Back in 1994, a small group of undergraduates toiling in a round, windowless room at the National Center of Computing Applications at the University of Illinois at Urbana-Champaign produced what was certainly the decade's, and perhaps the century's most revolutionary application: Mosaic. It's praises have by-and-large been duly sung, but one of its lesser known, and most remarkable attributes was its architecture: it is was what I call a "Big Bucket of Glue". The Mosaic application actually contained very little new code. It was, instead, constructed of a thin veneer of C adhesive code that bound together a rag-tag collection of existing applications. It was a harbinger of things to come.

With the increasing popularity of scripting languages with names beginning with letters like "P", this style/school of design has become one of our current era's most frequently deployed systems level architectures. Drawing on an increasingly broad, if not rich legacy of existing applications and components of all sizes, shapes and varieties, architects increasing slather on the glue, and ship these "Big Buckets of Glue" while their competitors are still inking high-road cartoons on their whiteboards.

While the "Big Ball of Mud" has remained the design of choice for application developers, the "Big Bucket of Glue" is increasingly the weapon of choice for system's level integrators. Indeed, the wildly popular Eclipse platform can be seen as a collection of black-box Java components bound together with reflective XML glue.

The End-to-End Principle and Programming Language Design

Earlier this year, I finally came across one of the greatest "classic" papers I'd never read: "End-to-End Arguments in System Design" by Saltzer, Reed, and Clark. The end-to-end argument was developed in the realm of internet protocol design. Succinctly put (and this format allows for no alternative) it argues that facilities, such as error checking, which one might be tempted to place at the lower levels of a system or network, might actually be redundant, and therefore useless as the lower levels, when they must be included in the upper levels of the system as well.

We argue that this argument applies as well to programming language design as it does to networking. For instance, facilities that some languages provide for security, protection, synchronization, namespace management, or type checking must often be re-implemented in applications programs in a variety of different ways, many at odds with the manner in which these facilities were provided in these underlying languages. Such duplication raises the question of why these now redundant facilities need be present in these programming languages at all.

This argument suggests a simpler, more dynamic approach to programming language design, based on an end-to-end analysis of the kinds of facilities users really use, may be better suited to the demands of our clustered, networked, threaded multi-core 21st world.

Seeds of Disquiet in Programming

Richard P. Gabriel, Sun Labs

Since the first days of computing there has been a disquiet about precision—or the machine. This disquiet implicates the nature of computational existence and how programs are treated. An early form of this disquiet was Bayesian inference which looks at probabilities as degrees of belief in contrast to purely deductive modes of reasoning. The trend in logic and mathematics continued with fuzzy sets and fuzzy logic which talk about set membership in vaguely defined sets. This early disquiet was with truth and certainty, both of which were shaken scientifically and philosophically by quantum mechanics where observables are characterized by probability distributions.

Later the disquiet became more focused and better defined: Expert systems were developed which worked with confidences—degrees of belief—and challenged programming. The challenge came in three parts: the sequence of steps to reach a conclusion should be dynamically synthesized rather than statically programmed; multiple lines of attack or reasoning can be pursued concurrently; and knowledge has primacy over technique. The impulse for an expert system developer when things don't work is to expand the knowledge base not reprogram procedures.

The essence of the disquiet has to do with what programs are and how they behave. The strong thrust in computing has been that programming is a precise encoding of a technique for solving a primarily mathematical problem. The disquiet is that not everything fits such a mold—the (very) execution of a program, its well being and robustness in the face of flaws, and the path it takes through the exigencies it faces while running seem to be of a different nature from that of simple computations and therefore require some other means, mechanisms, and ways of operating than do those more routine parts of the running program.

The disquiet has ebbed and flowed over the years, with neural nets, genetic programming, and even things like randomized algorithms on one hand, and ever more detailed and exhaustive type schemes, precise semantics, and methodological rigor on the opposite, as if one side wished to accommodate and survive the unexpected in processing while the other wished to control and dominate.

Finally, the disquiet may resolve to an approach that takes at least some part of computing as living while other parts are deathlike machines—the one suitable for inherently flexible and adaptive techniques and mechanisms, and the other, perhaps, remaining subject to rigor.

We the People

Andrew Hunt, The Pragmatic Programmers

Software development isn't about technology: it's about people. We are the raw material of software development, and that fact seems to have been lost in the continuing, hope-filled rush of new technology.

The two most important skills a software developer must have are the ability to communicate well and to learn quickly. We, the people, act as communication hubs: we have to communicate to the machine, using a variety of arcane syntactical constructs; to the end users, determining their wants, needs, and desires; and with our team members and the rest of our organization. Similar to a network router, we spend our days interpreting and retransmitting packets of ideas back and forth between these entities.

Similarly, we are constantly learning. Beyond the obvious learning of new and improved (sic) technology, we are learning subtleties of the problem domain, of the dynamics and quirks of the team, and the evolving characteristics of the system itself. We have to be skilled at rapidly evaluating, digesting, and applying ideas from a variety of sources, often in the face of incomplete or contradictory information.

The lucky undergraduate may have had a technical writing course, and perhaps some experience working a team environment, but that's probably all. We must do better: specific communication skills including facilitation, public speaking, interviewing techniques, and conflict resolution need to be addressed. Exposure to effective learning methods including summarizing techniques, speed-reading, and knowledge management/organizational techniques such as mind mapping need to be central to the developer's education; the specific technologies used for development (theories of computing, languages, design patterns, methodologies, and so on) remain transient, and thus secondary.

Software has become a pervasive and fundamental part of our society. It's time to move beyond producing computer scientists or mere programmers, and get serious about developing Software Developers.

Dear OOPSLA People,

Cristina Videira Lopes, UC Irvine

Let's face it: objects are here to stay. It's time to let them walk on their own and move on. Objects are just one of many great concepts in programming - other great concepts are waiting to be formulated. We need those fast. It's been 20 years since the first OOPSLA and 10 since Java came to be. What did we gain? A few good concepts and a few new programming tools. But developing software is still as hard as it ever was, if not harder. There's more "stuff" to learn, buzz words have multiplied like rabbits.

We must move out of this local maximum that we've been stuck in for so long. As the first step into the unknown, I dare you to drop those two little wheels on the side and have the courage to assume the substance that you've always had: the quest for better programming. Become PSLA.

Computer Science as Rhetoric and Performance

Brian Marick, Testing Foundations

When we think about how communication works, we usually think that we have knowledge in our head, we pack it into a message, we send that message over a communications channel to another person, who then unpacks it. That assumption is embedded in our very language, in sentences like "It's hard to put this idea into words." Michael Reddy called this the "conduit metaphor," and it's false.

It's better to think of communication as provoking a listener to assemble a meaning. Or, sometimes, to perform in a particular way without ever assembling a meaning.

Humans aren't much like computers receiving a packet, or even computers running a program.

Discarding the conduit metaphor has all sorts of implications, but I only have room to discuss the creation of program text. That text not only has to serve a flawless but unforgiving reader (some computer) but also a flawed but creative reader (any person). Because of the latter, we ought to study rhetoric, the art of persuasion, and literary criticism. (I'm especially fond of reader-response criticism.) But because we're the first in history to produce one text for two so radically different audiences, we also have immense scope for discovery, not just relearning.

I want to go a step further.

Programming is usually a group performance, one where a person's ongoing creation interweaves with everyone else's. What could we learn from a serious study of theatre, musical performance, improvisatory comedy, and group decision-making and brainstorming techniques?

Some Thoughts on Agency

Bonnie Nardi, UC Irvine

As machines become more intelligent and responsive, I find myself thinking about the nature of agency and how humans co-exist with other agents in the world. Table One categorizes different kinds of agents and agencies. Row 3 indicates that all agents can produce effects in the most general sense. Rows 4-5 indicate that when producing effects, some agents realize their own needs. Row 6 indicates that an agent may realize the intentions of (other) humans.

Living beings are a special kind of agent in that they strive to meet needs in the world, engaging other entities as they do so in a patterned way. An erupting volcano has tremendous agentic power, but it simply explodes, affecting whatever lies in its path without regard for where its cinders rain down, where its lava flows. By contrast, a plant with its biological needs reaches for the sun, it produces chlorophyll, its flowers attract bees of a particular kind, its seeds are eaten by certain birds who scatter them in the woods.

The category "non-human living beings (cultural)" includes organisms such as domestic animals, plants, and fungi; live vaccines; clones; and genetically engineered plants and animals. The distinction is between organisms that have evolved outside human intention and those that have been cultivated, cultured, husbanded, bred, cloned, or genetically modified. Producing effects, acting, and realizing intentions, while potentialities of certain kinds of agents, vary within the enactment of a specific activity, so we can consider three kinds of agency that may be present in any given situation:

Need-based Agency. Human beings have both biological and cultural needs. To meet their needs, they form intentions and act from the intentions. Similar types of agency are manifested by social entities (though they do not have biological needs) and higher animals (though they do not have cultural needs).

Delegated agency. Various things and living beings can be said to realize intentions, but intentions delegated to them by somebody or something else. These things and living beings are agents in the sense of acting on somebody else's behalf.

Conditional agency. Anything and anyone can produce unintended effects. Even without having a need or intention, something or somebody may constitute a force—or condition—to be reckoned with.

The table suggests an interesting tension. It is common human practice to design new technologies, especially today. But every agent is capable of producing effects for which there is no intention. The more cultural things (living and non-living) we have in the world, based on human design and intention, the more possibilities we introduce for conditional agency, that is, for new kinds of unintended effects.

Computational Diversity, Practice and a Passion for Applications

Dave Thomas, Bedarra

THE FALLACY OF THE "RIGHT" THING: The myopic industrial context that is being imposed on many of our best educators and their students is distressing. There is a misleading assumption that software engineering is simply a matter of knowing how to use the latest "right" technology.

PROMOTING COMPUTATIONAL DIVERSITY - OBJECTS ARE NOT EVERYTHING: Students need to see beyond OO and a particular OO technology. Computer Science is not just about objects, just as science is not just about chemistry and business is not just about accounting. Students need an appreciation and understanding of computational diversity. Different tools, techniques, and metaphors are not concepts that should only be offered in optional courses or limited to graduate students. It is not sufficient to make passing reference to this a survey course.

COOPERATIVE EDUCATION CONSIDERED ESSENTIAL: Students need to learn how to work with designs, specifications, and implementations produced by other people.

A PASSION FOR APPLICATIONS—COMPUTATION IN CONTEXT: Students need to see objects in context. Students should be provided with exercises based on an actual application that can provide an opportunity to provide a concrete substrate for abstract concepts.

REQUISITE VARIETY IN CS FACULTY: In the early days of computer science the faculties contained a rich diversity of scientists, engineers, philosophers, and even talented people without a PHD! Each of these brought their external experience context to the department and to the students. Today computer science departments are so pure that if they were a liquid the lighting through them would not be refracted into the beautiful rainbow caused by the impurities.

THE CHALLENGE OF BEING INDUSTRIALLY CURRENT: Many faculty have moved away from computation simply because they feel incapable of keeping up with what is happening in the industry. Of course they see IT professionals struggle to keep on top of the latest thing. Those of us associated with such innovations need to find ways to keep our educational colleagues current, by helping them understand the essence of each commercial wave without forcing them to crawl through the plethora of quick to market books, tools and APIs. We need to make an effort to show how this is similar but different from what came before, rather than the marketing departments claims of our latest "New New Thing" being new and completely different.

SUMMARY: We need to ensure that students—the potential software industry leaders of the future—are exposed to the concept of computational diversity. They need to critically assess ideas and products that are commonly perceived as being "in", while understanding the potential relevance and utility of ideas and products that are "out". A wide spectrum education will give students the knowledge to make technical decisions based on what solution best applies in a given situation.

Humanity is Half of the Changes in Gaia's Kyberos

Rob Tow, Sun Labs

Life has existed on Earth for 3.5G years. During that time, eight basic changes in cybernetics have occurred in the organization of life, where a higher level of kyberos seen as perception-representation-action loops are layered on top of the previous system. Four of these changes are expressed by humanity, and two of those are contemporary to the Now. Using the language of invention and teleology, these are:

(1) The basic archea cell

(2) The merger of bacterial cells into the eucharotic cell - e.g, mitochondria, etc

(3) Multicellular organisms.

(4) Nervous systems - where a temporary representation is created to organize behavior on the time scale of the life of the individual; it is lost to entropy when the individual dies.

These 1st four system levels progress by neo-Darwinian evolution, modified by symbiogeneis.

(5) Symbolic speech and language - now the representations in (4) above can be transmitted from one nervous system to another - memes come into existence, along with narrative intelligence.

Memes progress by Lamarkian evolution.

(6) The exteriorization of memes in written form - the oldest known such are 30K year old calendars

(7) The making of Turing complete meme engines, that have their own perception-representation-action loop - computers.

(8) The use of computers (and other instrumentalities) to connect the agency of narrative intelligence - memes - to DNA - thus connecting the highest level of Lamarkian evolution to intervene into the neo-Darwinian and symbiogentic processes - the biggest cybernetic change in the history of life on earth.

What is the least understood thing in computing that if only it were understood, everything would change?

Glen Vanderburg, Countrywide Home Loans

We have a mania for abstraction, but our brains are optimized for the concrete. We are trained to abstract, to generalize. Butler Lampson famously said "All problems in Computer Science can be solved by another level of indirection." The drive for abstraction isn't wrong; it's an essential part of software development. But we generally do it so irregularly and so poorly, and one reason is that abstraction is somewhat unnatural for us.

We learn best when we can hear, taste, smell, and (especially) touch and manipulate the world. The best programmers are good at a kind of "mental modeling" that allows them to do the same with the abstract components of our field. A good spatial sense and visualization seems to accompany software skill. We need to find better ways to bring the rest of our brains—the parts that are designed for life in the real world—to bear on the problems of programming. We need to build better tools, tools that don't lock these complex, multidimensional constructs into oversimplified, two-dimensional representations. More importantly, we need to teach and cultivate those mental mapping and modeling skills that the great programmers use instinctively, independent of their tools. We need to program with our whole selves, not just the tiny parts of our brain that are good at abstraction.

Onward! Keynote: The End of Users

Wednesday, 8:30-10:00 ***Town & Country Room***

Mary Beth Rosson, Professor, Pennsylvania State University

Over the past 20 years, user interface designers and usability engineers have studied and refined human-computer interaction techniques with the goal of improving people's productivity and experience. But the target of these efforts "the end-user" is fast becoming a thing of the past. Many people now construct software on their own, building artifacts that range from email filters to spreadsheet simulations to interactive web applications. These individuals are use-developers: they build ad hoc solutions to everyday computing needs.

Will use-developers help to resolve the software crisis? Given the right tools, people and groups may be able to rapidly develop custom solutions to many context-specific computing requirements, eliminating the wait for IT professionals to analyze and engineer a solution. Or are these individuals a danger to society? Use-developers are informal programmers with no training in software construction methods or computing paradigms. They have little intrinsic motivation to test their products for even basic concerns like correctness or safety. In this talk I argue that the transformation of end-user to use-developer is well underway and discuss the prospects for maximizing the benefits to society while addressing the risks.

Mary Beth Rosson is Professor of Information Sciences and Technology at The Pennsylvania State University. She received a PhD in experimental psychology in 1982 from the University of Texas. Prior to joining the new School of Information Sciences and Technology at Penn State in 2003, she was a professor of computer science at Virginia Tech for 10 years and a research staff member and manager at IBM's T. J. Watson Research Center for 11 years. Rosson was among the earliest researchers to study the psychological issues associated with the object-oriented paradigm, and spent many years developing and evaluating object-oriented tools and training for professional programmers. One of her abiding interests has been the interplay between the concerns of human-computer interaction and software engineering. Recently she has been studying the tools and practices of end-user developers in educational and general business contexts.

Rosson has participated in the OOPSLA conference in many capacities, as a member of the technical program committee as well as the conference committee; she introduced the OOPSLA Educators' Symposium in 1992, the Doctoral Consortium in 1994, and served as General Chair for OOPSLA 2000. She is also active in the ACM SIGCHI community, where she is General Chair for CHI 2007. Rosson is author of Usability Engineering: Scenario-Based Development of Human-Computer Interaction (Morgan Kaufmann, 2002) as well as numerous articles, book chapters, and tutorials. More information is available at http://ist.psu.edu/rosson.

Onward Presentations 2: Ambient and Organic

Wednesday, 10:30-12:00 ***Town & Country Room***

Ambient Oriented Programming

Jessie Dedecker, Vrije Universiteit Brussel
Tom Van Cutsem, Vrije Universiteit Brussel
Stijn Mostinckx, Vrije Universiteit Brussel
Theo D'Hondt, Vrije Universiteit Brussel
Wolfgang De Meuter, Vrije Universiteit Brussel

Ercatons and Organic Programming: Say Good-Bye to Planned Economy

Oliver Imbusch, Living Pages Research GmbH
Guido von Walter, Living Pages Research GmbH
Falk Langhammer, Living Pages Research GmbH

Organic programming (OP) is our proposed and already emerging programming model which overcomes some of the limitations of current practice in software development in general and of object-oriented programming (OOP) in particular. Ercatons provide an implementation of the model. In some respects, OP is less than a (new) programming language, in others, it is more. An "ercato machine" implements the ideas discussed and has been used to validate the concepts described here.

Organic programming is centered around the concept of a true "Thing". A thing in an executing software system is bound to behave the way an autonomous object does in our real world, or like a cell does in an organism. Software objects do not. Therefore, traditional software systems must be planned ahead like in a centrally planned economy while with OP, software systems grow. This fact is traced back to be the root why current software development often fails to meet our expectations when it comes to large-scale projects. OP should then be able to provide the means to make software development achieve what other engineering disciplines have achieved a long time ago: that project effort scales sub-linearly with size.

With OP we introduce a new term because we hope that the approach we are pursuing is radical enough to justify this.

Working with Vision

Wednesday, 13:30-15:00 ***Town & Country Room***

Organizers of Onward! style workshops discuss approaches to innovation and inspiration in breaking new ground.

Chair: Steven Fraser

Scrapheap Challenge - A Workshop in Post-Modern Programming
Ivan Moore, Nat Pryce

Apprenticeship Pedagogical Patterns
Pam Rostal, David West, Joseph Bergin, Jutta Eckstein, Mary Lynn Manns, Linda Rising

The Future of Data-Centric and Information-Centric Computing
Steve Lakin, Yvonne Coady, Jeff Eastman, Elena Gaura, Sarah Mount, Bob Newman

Extravagaria III: Hunting Creativity
Richard Gabriel, John Gribble, Brenda Laurel

Croquet: A platform for Collaboration
David Smith, Alan Kay, Julian Lombardi, Mark McCahill, Rick McGeer, Andreas Raab, David P. Reed

Onward Presentations 3: Structure and Semantics

Wednesday, 15:30-17:00 ***Town & Country Room***

Living Structure and the Software Garden
Russell Holt, holt-research

Complex structures cannot be built; they must grow. Natural living systems grow and are the products and creators of the environment; nothing exists in isolation. To create software with the complexity of natural forms, we are going to have to figure out how to grow our software as well. This begins with changing the way we represent it so that changes are graceful and fundamentally stable. One way to do it, which I am developing, is to represent software as a connected web of relationships among the parts. If this representation can be integrated with its environment - that is, all the information that surrounds the creation of the software, all the more likely the system as a whole will reflect reality and pave the way for automated processes to work together with humans in the same environment.

DOSC: Dispersed Operating System Computing
Ramesh Karne, Towson University
Tufail Ahmed, Towson University
Karthick Jaganathan, Towson University
Nelson Rosa Jr, Dartmouth College

A Delta-driven Execution Model for Semantic Computing
Roly Perera, Dynamic Aspects
György Koch, Dynamic Aspects
Jeff Foster, Dynamic Aspects

We describe (and demonstrate) the execution model of a computing platform where computation is both incremental and data-driven. We call such an approach delta-driven. The platform is intended as a delivery vehicle for semantically integrated software, and thus lends itself to the semantic web, domain-driven development, and next-generation software development environments. Execution is transparent, versioned, and persistent. This technology - still at an early stage - is called domain/object.

The execution model of domain/object is a radical departure from that of most programming languages and virtual machines in use today. Execution takes place solely by the propagation and interpretation of object deltas. When a data value changes, a description of that change in the form of a delta propagates to all dependent data values. The incoming delta is interpreted as a (possibly null) delta in the dependent value, which in turn recursively propagates to its dependents. When changes propagate across domains, the delta translation process is an interpreter in the traditional computer science sense, translating actions in the “source” language as they are received into actions in a “target” language and applying them to the target domain. (Action and delta are synonymous in our nomenclature.)

Under the delta-driven approach data values are “live”, not snapshots like variables are in traditional programming languages. Abstractly the model is more similar to a spreadsheet than a standard programming language. Domain/object is thus a kind of dataflow language.

The “integration of disparate systems” is the number one priority facing internet-centric businesses today, according to IBM. Yet as an industry we have spent precious little time understanding how to make software really connect. While much remains to be done, domain/object represents an important move in the direction of truly semantic computing.

Onward! Films

Thursday, 10:30-12:00 ***Town & Country Room***

Onward! films investigate, in a visually appealing form, ideas, concepts, or insights related to programming.

Agile Workspaces... For the rest of us

Dean Mackie, Ontario Teachers Pension Plan, Canada
Jason Rogers, Ontario Teachers Pension Plan, Canada
Gifford Louie, Ontario Teachers Pension Plan, Canada
Niall Shaw, Ontario Teachers Pension Plan, Canada

This film tells the story of the introduction of furniture on wheels into a very traditional corporate culture. We've all heard of innovative office environments used by design firms, startups, and skunkworks, but what about the rest of us? At a 90-year-old $80-Billion financial institution we attempted to implement and measure the effects of an office environment that would be a logical extension of our existing culture, and would also better support collaborative work and the use of Agile Software Development methodologies.

A cross-functional project team received new office furniture on wheels which allowed more team interaction and fast layout reconfiguration. The economics and corporate culture effects of this move were recorded. The team was surveyed on ergonomics and ability to collaborate six months before and six months after the change, and the results compared with a team that did not receive the change over the same period. While the sample size was too small to imply universal results, it did anecdotally indicate the benefits of continuing and expanding the implementation.

The Digital Tipping Point

Christian Einfeldt **Holden Aust** **Diane Mackay** **Justin Flint**
Adam Doxtater **Alexandro Colorado** **Danese Cooper** **Josh Berkus**
Dorothee Weiler **Lars Noodén** **Dominik Hierl** **Cooper Stevenson**
Doris Waizmann **Linda Worthington** **Ben Horst** **Kass Stevenson**
Paul Donahue **Ursula Schmidt** **Sky Christopherson** **Rufus Laggren**

You can't buy love, and you can't buy freedom, but you can share both. And free open source software is gonna help you do both. The crew of the Digital Tipping Point traveled to Brazil, Spain, German, Scotland and (gasp) Oregon to find out how free open source software and the Internet, like the printing press, is going to change how human beings express themselves, how they organize themselves, and how they govern themselves. Here at OOPSLA, we are presenting a raw, 9 minute clip of interviews with some of the thought leaders as they weigh in on how open source is succeeding on the desktop and elsewhere, and what's at stake in the battle for freedom in cyberspace.

Parental Inheritance

Morten Telling Nielsen, University of Southern Denmark, Denmark

The film postulates the existence of a third kind of relationship between classes in strongly typed languages. The film takes offset in problems modeling some special situations, where both specialization and aggregation seems to fail (composition and association are here considered to be subtypes of aggregation, since they all are "has a" relationships and they all work by one object having a reference variable to another object). I have named the relationship type "Parental inheritance". Parental inheritance is not a classic superclass/subclass relationship or an aggregation; it is an entity in its own right, with its own rules and ability to model the real world. Parental inheritance have features in common with both aggregation and specialization, but are closer related to the relationship between the static and the nonstatic contexts in languages like C++ or Java. The relationship also has similarities with tuppels of tables in relational databases that have foreign key references.

The Crisis in Systems Code Maintenance: Sourceforge, we have a problem

Rebeca Dunn-Krahn, University of Victoria, Canada
Yvonne Coady, University of Victoria, Canada
Jessica Maple, University of Victoria, Canada

Linguistic support for modern programming paradigms has not been welcomed into most of today's mainstream operating systems. Linus Torvalds has decreed that Linux will never again entertain C++, "In fact, in Linux we did try C++ once already, back in 1992. It sucks. Trust me, writing kernel code in C++ is a BLOODY STUPID IDEA...". Similarly, Pantelis Antoniou, an embedded PowerPC kernel developer, has captured popular systems-sentiment about aspect-orientation, "People like to live in denial; thinking that programming shouldn't be this hard right? There must an easier way, if only those pesky developers followed $fashionable_methodology_of_the_day...". As a consequence, though a number of systems have been progressively restructuring services to leverage higher-level paradigms, it is intentionally done without language support. This decoupling of paradigms and language mechanisms appears to suggest that conventional wisdom in the systems community prejudices modern programming methodologies because they may impose unnecessarily heavyweight manifestations of paradigms, and pollute otherwise elegant and optimized hand-crafted C code. Simply put, if it ain't broke, don't fix it. We believe that there is a growing body of evidence to suggest that, if the systems community continues to refuse support for a paradigm shift, system evolution will slow down to an unacceptable level. Already, valuable code is not being integrated into systems in a timely fashion because the tools meant to facilitate this can, and often do, impede the process. Simply put, it is broke and we believe we know how to fix it.

Panel: Yoshimi Battles the Pink Robots
[Programmers v. Users: The Culture War]

Thursday, 13:30-15:00 ***Town & Country Room***

Please see the description on page 48 in the Panel section.

Chair: Steve Berczuk, Iron Mountain Inc.

OOPSLA panels have consistently been among the best-attended and well-received attractions at the conference. These panels offer an engaging, entertaining and informative examination of timely topics from a variety of viewpoints. OOPSLA panels offer a unique forum to spotlight emerging issues. They also give the OOPSLA community a way to tackle controversial and divisive topics head-on in a fun, interactive way that can shed welcome light on the issues we all must deal with. We hope that you find these panels informative, memorable, and entertaining.

Aspects: Passing Fad or New Foundation?

Tuesday, 13:30-15:00 ***Sunrise***

ABSTRACT: Aspect-oriented software development (AOSD) has a lot of interest in the research community. It has also found early adopters in application development and middleware. This panel discusses the potential expansion and use of AOP into mainstream software development.

This question is not just directed to the aspect research community, but also to practicing software development teams and organizations. The panel will explore the appropriate position and awareness of aspect-orientation amidst other advances in software engineering; how to prepare organizations for adoption; and what additional research and development is necessary. The panel discussion will help the OO community to understand the appropriate use of aspect-orientation. It will also highlight areas where additional efforts by AOSD researchers and professionals are required.

CHAIR: Steve Berczuk, Iron Mountain

PANELISTS:

Matthew Webster, IBM

Matthew Webster joined IBM in 1989 with a degree in Physics with Computer Science from Southampton University and since then has worked on a number of software projects at the Hursley lab. He moved to the Java Technology Centre in 1997 initially as a technology evangelist then working on the restructure of the IBM JVM and leading the development of advanced Garbage Collection and Class Loading features. Matthew is a senior software engineer developing AOP technology for use in IBM software products and is co-author of a book on AspectJ and Eclipse published this year.

Jack Greenfield, Microsoft Corporation

Jack Greenfield is an Architect for Enterprise Frameworks and Tools at Microsoft. A well known speaker and writer, he is a co-author of the book "Software Factories: Assembling Applications with Patterns, Models, Frameworks and Tools. He has also contributed to UML, J2EE and related OMG and JSP specifications.

Jack thinks AOSD is a good idea expressed at the wrong level of granularity. He sees value in recognizing cross cutting concerns and specifying software independently from multiple interrelated viewpoints, but he sees raw source code weaving as throwing away lessons learned the hard way about the value of encapsulation. He would rather use aspects to separate concerns across the software life cycle, as a basis for specifying software architectures in computationally useful ways, for delivering architectural guidance to developers in context, and for the manual and automatic enactment of architectural guidance, as demonstrated by software factories.

Ivar Jacobson, Ivar Jacobson International & Jaczone

Ivar Jacobson is a pioneer of components and component architecture, use cases, the Unified Modelling Language, the Rational Unified Process and lately aspect-orientation with use cases. He is the founder of Jaczone, developing intelligent agents for software development, and of Ivar Jacobson Consulting, training and mentoring in software best practices including aspect technology.

Ivar believes that use-case driven development and AOP complement each other very well. AOP is the missing link in use-case driven development, and use-case driven development provides the guidelines for proper application of aspects. Use cases are early aspects and are keys to effectively separate concerns. Application use cases keep normal (non-crosscutting) concerns separate, whereas infrastructure use cases keep crosscutting concerns separate. The combined adoption of use cases and aspects, will have dramatic impact on the cost, quality and other measures of software.

Gregor Kiczales, University of British Columbia

Gregor Kiczales is Professor of Computer Science at the University of British Columbia where he engages in research on aspect-oriented programming (AOP) and other technologies that help programmers make the code look like the design. While at Xerox PARC, he led the teams that developed aspect-oriented programming and AspectJ.

Gregor believes that the reason AOP is being adopted so quickly is that its an idea who's time has come - developers are well aware of modularity problems caused by crosscutting concerns. AOP gives them a tool to solve those problems, and make code more modular. In doing so it makes software more valuable, and programming more fun. But as we move into the next stages of adoption, there are many problems to solve, including better training, tooling, libraries, business cases and standardization.

Dave Thomas, Bedarra Research Labs, Carleton University, University of Queensland

Dave Thomas is chairman of Bedarra Research Labs. He is an adjunct research professor at both Carleton University and University of Queensland, on the editorial board of the Journal of Object Technology and Transactions on AOSD.

Dave thinks that AOP is a very promising systems programming technology but is not yet convinced where and if it is something that belongs in the toolset of every analyst and developer. Aspect libraries appear to be a promising way to bring the benefits to main stream. Aspects have been useful for restructuring poorly designed middleware, but it isn't clear how much they contribute to avoiding such messes in the first place. Recent research in AOP, such as FOP suggests their may be even simpler ways to compose programs without embedding aspects in the programming languages.

Fostering Software Robustness in an Increasingly Hostile World

Wednesday, 10:30-12:00 ***Sunrise***

ABSTRACT: Software can kill. What are you doing to stay alive? Our world faces an increasingly hostile environment with challenges in complexity, technology, social engineering and clashing cultures. Failure to achieve sufficient software robustness can lead to customer dissatisfaction, financial loss, or in extreme cases loss of life. This panel brings together differing contexts and solution approaches.

CHAIR: Steven Fraser, QUALCOMM

Steven is a senior member of staff at QUALCOMM's Learning Center in San Diego California. From 2003 to 2004 he was an independent consultant in Santa Clara California. Previous to 2002 Steven held a variety of diverse software technology program management roles at Nortel Networks including: Process Architect, Senior Manager (Disruptive Technology and Global External Research), and Process Engineering Advisor. In 1994 he spent a year as a Visiting Scientist at the Software Engineering Institute (SEI) collaborating with the Application of Software Models project on the development of team-based domain analysis techniques. Steven holds a PhD in EE from McGill University in Montreal, Canada. Steven is an avid operatunist and videographer.

PANELISTS:

Djenana Campara, Klocwork

Djenana Campara is the founder and CTO of Klocwork a company devoted to proving automated solutions to facilitate the development and delivery of reliable, secure, and high-performance software systems. Prior to founding Klocwork, Djenana was a manager and systems architect at Nortel Networks. Djenana co-chairs the OMG Architecture-Driven Modernization Special Interest Group and serves as a board member on the Canadian Consortium of Software Engineering Research (CCSERC). Djenana holds a Bachelor of Science degree in electrical engineering and computer science from the University of Sarajevo.

Carl Chilley, Xansa

Carl Chilley is an Executive Consultant with Xansa in the UK and primarily focuses on business architecture across all vertical and horizontal market sectors. Carl has been in the industry for over 25 years, having been the European software project manager for the iconic Xerox Star development, working his way through various architectural forms and styles in pursuit of the delivery of usable and viable business solutions. Most recently Carl has been applying the precepts and concepts of service-driven architecture in the development of solutions for his clients. Noting that Xansa has a market-leading Indian capability and a solid reputation for both outsourcing and off-shoring process and IT systems, Carl has also been looking at how to best integrate the dispersed capabilities of organizations to effect effective delivery in a cost effective manner.

Richard Gabriel, Sun Microsystems

Richard Gabriel (Ph.D., MFA) is a Distinguished Engineer and chief scientist of a small laboratory at Sun Microsystems, researching the architecture, design, and implementation of extraordinarily large systems as well as development techniques for building them. He is Sun's open source expert, advising the company on community-based strategies. He is also President of the Hillside Group, a nonprofit that nurtures the software patterns community by holding conferences, publishing books, and awarding scholarships.

Ricardo Lopez, QUALCOMM

Ricardo Lopez is a Principal Engineer in the Office of the Chief Scientist at Qualcomm. He is responsible for Software Architecture, Software Process & Methodology, and sometime Just Plain Old Software (JPOS) whenever the need arises. Architecting & Designing Software for 30 years, he has been an evangelist for OO technology for the last 18 years and he has the arrow heads to prove it. He continues to advance and foster quality software architectures, designs and practices wherever he goes.

Software in today's world requires a deep understanding of the increasing dependence that civilization has placed upon this uniquely human artifact; Software development begs improved sensitivity to the inherent complexities naturally encountered as we expand the horizons of our software into ubiquitous computing and all the increased leverage that will entail; and it demands a commitment to quality, security and trust not found currently on the campuses of our largest software producers. The world we are moving into can not and will not tolerate or survive "Good Enough".

Dave Thomas, Bedarra

Dave Thomas is founder of Bedarra Research Labs and adjunct research professor at Carleton University and the University of Queensland. Bedarra current focus areas are eLearning; Next Generation Application Development; Agile Software Development and Pervasive Computing. Dave is well known to the object community as the founder of Object Technology International (OTI) developers of Envy-Developer a unique CM environment for object oriented development; virtual machines for Smalltalk, Java and IDEs for IBM VisualAge for Smalltalk; for Java; Micro Edition for Embedded Systems and Eclipse. Bedarra focuses on the transfer of research from the lab to industry. Dave is a founding director in Agile Alliance, and columnist for the Journal of Object Technology (JOT) and OT Land.

Greg Utas, Pentennea

Greg Utas obtained an Honors BSc in Computer Science from the University of Western Ontario (Canada). In 1981 he joined Bell Northern Research (the research arm of Northern Telecom—later to become Nortel Networks), where he served as the principal software architect for various switching products. As the Chief Architect of GSM Core networks, Greg let a team of 50 designers who redesigned the product's call processing software using object-oriented techniques. For this work, he received Nortel Technology Award for Innovation and became the first software architect at Nortel's Director level. In March 2002, Greg joined Sonim Technologies as the Chief Software Architect, responsible for the design of push-to-talk services for wireless networks. He left Sonim in May 2004 to become a consultant specializing in carrier-grade software. Greg is the recent author of "Robust Communications Software" (Wiley, January 2005).

The Agile Panel

Wednesday, 13:30-15:00 ***Golden West Room***

ABSTRACT: Any ordinary panel member on an ordinary panel can sit up front and spout the same old stuff. The panelists for this event must be agile in the strictest sense of the word. Why? Because they will be asked to take a random position in response to questions from both the moderators and the audience. They must present their arguments in a timed two-minute speech or present a rebuttal of the initial points in the same restricted two-minute interval. The assignment to the position will be made by the moderators. Don't miss this one!
Keywords: hot topics, debate

CHAIR: Linda Rising, Independent consultant

Linda Rising has a Ph.D. from Arizona State University in the area of object-based design metrics. Her background includes university teaching experience as well as work in industry in the areas of telecommunications, avionics, and strategic weapons systems. She has been working with object technologies since 1983. She is the editor of A Patterns Handbook, The Pattern Almanac 2000, and Design Patterns in Communication Systems. She has a number of publications including: "Patterns for Collaboration," Cutter IT Journal, February 2005. This and other articles are available on her web site: www.lindarising.org. She is a regular contributor to the DDC-I Online Newsletter: ddci.com/newslatest_news_archive.shtml She has presented a number of tutorials and workshops at JAOO, OOPSLA, and other conferences.

CHAIR: Mary Lynn Manns, University of North Carolina at Asheville

Mary Lynn Manns earned her PhD from De Montfort University in England in the area of introducing software patterns into organizations. Her 20-year teaching experience in academia and industry includes an outstanding teaching award in 1995. She is currently at the University of North Carolina at Asheville where she has taught courses in computer science and management information systems. She is the co-author, with Linda Rising, of the popular book on introducing new ideas to teams and organizations: Fearless Change: Patterns for Introducing New Ideas. Mary Lynn has done numerous presentations on the topic of introducing new ideas into organizations.

PANELISTS:

Angela Martin, Victoria University of Wellington

Angela has over ten years of wide ranging information systems experience and has a firm grounding in all aspects of systems integration and development. She is a PhD Candidate at Victoria University of Wellington, New Zealand, supervised by James Noble and Robert Biddle. Her PhD research utilizes in-depth case studies of the XP Customer Role, on a wide range of projects across Australasia, the United States and Europe. She is a board member of the Agile Alliance and has presented at XP2003, OOPSLA 2003 Onward!, XP2004, ADC 2004 and will be offering a tutorial on the XP Customer at OOPSLA 2005.

Kevlin Henney, Curbralan

Kevlin Henney is an independent consultant and trainer based in the UK. The focus of his work is in programming languages, OO, UML, agile development, patterns, and software architecture. At various times he has been a regular and irregular columnist for C/C++ Users Journal (online), Application Development Advisor (UK), JavaSpektrum (Germany), Java Report, and C++ Report. He is also on the advisory board for Hillside Europe, has been involved with the committees of various conferences and language standards ("a committee is a cul-de-sac down which ideas are lured and then quietly strangled"), and is a popular speaker at conferences in North America and Europe.

Alan O'Callaghan, De Montfort University

Alan O'Callaghan is a Senior Lecturer in the Faculty of Computing Sciences and Engineering at De Montfort University in the UK, as well as a researcher and a practicing consultant. Allan has been at DMU for the last sixteen years after working for large companies such as the London Underground, the British Oxygen Company and Kodak UK. He has authored two books and published more than 40 articles and papers. He has also written two pattern languages. But his real passions are Guinness, the 'craic' and Chelsea Football Club!

Rebecca Wirfs-Brock, Wirfs-Brock Associates

Rebecca is the inventor of Responsibility-Driven Design and lead author of two books: Object Design: Roles, Responsibilities, and Collaborations (2003), and Designing Object-Oriented Software (1990). She is a board member of the Agile Alliance and an OOPSLA lifer having attended all 20 conferences! Rebecca has been involved with object technology since its infancy. Among her widely-used inventions are use case conversations and object role stereotypes. From development on the Tektronix implementation of Smalltalk in the early 1980's, through years of development and training experience, she is recognized as an innovative and influential practitioner of object-oriented design.

A friend recently sent me a link to the World's Smallest Political Quiz. After answering 10 personal value and economic questions, you are rated a Liberal, Centrist, Conservative, Libertarian, or Statist. Depending on my mood I'm a liberal or a centrist. I lean towards supporting personal liberties but believe in some governmental control especially for the environment. This got me thinking about a similar test for software folks. Do you lean towards agile, centrist, traditional, anarchic, or formal software development practices? If so, what values do you hold? We need to understand what drives process differences as well as what values we hold in common. Agilists and traditionalists both value process. Anarchists don't—whatever works, works. Neither do formalists—they value formalisms over any process. Yet there are differences. Agilists value reactive planning more; traditionalists prefer predictive planning. Yet most want to eliminate defects sooner and could benefit from Test Driven Development. You'll be at the top of your game if you look for practices that support your values regardless of their origin. In this belief I am a centrist and an opportunist.

Echoes: Structured Design and Modern Software Practices

Thursday, 10:30-12:00 ***Golden West Room***

ABSTRACT: It has been 30 years since Structured Design first evolved. The Software Engineering Institute's website [March 2005] describes Structured Design as a "traditional approach" that does not lend itself well to object orientation. This panel brings together software design visionaries to discuss and debate "echoes" in software practice.

CHAIR: Steven Fraser, QUALCOMM

Steven Fraser is a senior member of staff at QUALCOMM's Learning Center in San Diego. Between 2002 and 2004 Steven was an independent consultant. From 1987 to 2002 Steven held a variety of diverse software technology program management roles at Nortel Networks including: Process Architect, Senior Manager (Disruptive Technology and Global External Research), and Process Engineering Advisor. In 1994 he spent a year as a Visiting Scientist at the Software Engineering Institute (SEI) collaborating with the Application of Software Models project on the development of team-based domain analysis techniques. Steven completed his doctoral studies in Electrical Engineering at McGill University in Montreal, Canada. Steven is an avid operatunist and videographer.

PANELISTS:

Kent Beck, Three Rivers Institute

Kent Beck is the founder and director of the Three Rivers Institute (TRI). Following work by Jim Coplien and Ward Cunningham on software development process, with Ron Jeffries and the C3 team at Chrysler he invented and named Extreme Programming, resulting in the Jolt Productivity Award-winning Extreme Programming Explained: Embrace Change (now in its second edition). He is the co-author of Planning Extreme Programming with Martin Fowler, with whom he also collaborated on Refactoring: Improving the Design of Existing Systems. With Ward Cunningham he wrote HotDraw, a widely copied drawing editor framework, pioneered patterns for software development, and popularized CRC cards. He channeled the Ancient Smalltalk Masters to produce The Smalltalk Best Practice Patterns, and revived a decades-old technique in Test-Driven Development By Example. He lives in rural Oregon on a 20 acre farm with a dwindling but still impressive gaggle of children, his lovely wife Cindee, and a flock of chickens.

Grady Booch, IBM

Grady Booch is recognized internationally for his work on software architecture, modeling, and software engineering. An IBM Fellow, ACM Fellow, World Technology Network Fellow, and Software Development Forum Visionary, Booch has lectured and consulted worldwide. He is a member of the Association for Computing Machinery (ACM), the Institute of Electrical and Electronics Engineers (IEEE), the American Association for the Advancement of Science (AAAS), and Computer Professionals for Social Responsibility (CPSR). Booch has been with IBM Rational as its Chief Scientist since Rational's founding in 1981. Booch is one of the original developers of the Unified Modeling Language (UML), and is the author of six best-selling books, including the UML Users Guide and the seminal Object-Oriented Analysis with Applications. He has published several hundred technical articles on software engineering, including papers published in the early '80s that originated the term and practice of object-oriented design. Booch received his bachelor of science from the United States Air Force Academy in 1977 and his master of science in electrical engineering from the University of California at Santa Barbara in 1979.

Larry Constantine, Constantine & Lockwood

Larry Constantine, often called the father of structured design and analysis, is one of the pioneers who helped construct the foundations of modern software engineering theory and practice. He is the originator of such widely used models as dataflow diagrams. He devised one of the first notations for modeling the architecture of software and introduced notational conventions ultimately reflected in modern object modeling techniques. He invented the metrics of coupling and cohesion and the underlying theory of program complexity on which they are based. In recent years his attention has turned to interaction design and techniques for enhancing user performance. He is himself an award-winning designer with patents in human-machine interaction to his credit. With Lucy Lockwood, he invented essential use cases, regarded by many as a best practice in user requirements modeling, along with usage-centered design, the widely practiced model-driven process based on them. He has published more than 175 papers and 17 books in both the human sciences and information sciences, including the classic text, Structured Design, written with Ed Yourdon. With Lucy Lockwood he wrote Software for Use, winner of the Jolt Award as best book of 1999. His papers have been widely reprinted and his books have been translated into nine languages. In wide demand as a presenter and teacher, he has taught in 18 countries and has keynoted numerous international conferences. He is chief scientist at Constantine & Lockwood, Ltd., the international design and consulting firm he co-founded, and is a former professor of information technology at the University of technology, Sydney (Australia).

Brian Henderson-Sellers, University of Technology, Sydney

Brian Henderson-Sellers is Professor of Information Systems, Director of the Centre for Object Technology Applications and Research (COTAR) at the University of Technology, Sydney and a member of the Department of Software Engineering. He is author of numerous papers including eleven books on object technology and is well-known (MOSES, COMMA and OPEN) and in OO metrics. More recently, he has become involved in the application of method engineering concepts to agent-oriented methodology construction. Brian is on the editorial board of the Journal of Object Technology and Software and Systems Modeling and was for many years the Regional Editor of Object-Oriented Systems, a member of the editorial board of Object Magazine/Component Strategies and Object Expert. In 1990, he founded the Object-Oriented Special Interest Group of the Australian Computer Society (NSW Branch) and was Chairman of the Computerworld Object Developers' Awards committee for ObjectWorld 94 and 95 (Sydney). He is co-founder and leader of the international OPEN Consortium. He is a frequent, invited speaker at international OT conferences, and, in 1999, he was voted number 3 in the Who's Who of Object Technology (Handbook of Object Technology, CRC Press, Appendix N). Brian's current research projects include OO and AO modeling (particularly aggregation in UML and OML), OO Process (OPEN), AO methodology construction, organizational transition to OO, OO metrics (including requirements and complexity metrics), OO ontologies and component-based development. He was recently a member of the Review Panel for the OMG's Software Process Engineering Model (SPEM) and UML 2.0 standards initiatives. In July 2001, Professor Henderson-Sellers was awarded a Doctor of Science (DSc) from the University of London for his research contributions in object-oriented methodologies.

Steve McConnell, Construx

Steve McConnell is CEO and Chief Software Engineer at Construx Software where he writes books and articles, teaches classes, and oversees Construx's software engineering practices. Steve is the author of Code Complete (1993, 2004) and Rapid Development (1996), both winners of Software Development magazine's Jolt award for outstanding software development books of their respective years. In 1998, he published Software Project Survival Guide and in 2004 he published Professional Software Development (2004). Steve has worked in the desktop software industry since 1984 and has expertise in rapid development methodologies, project estimation, software construction practices, performance tuning, system integration, and third-party contract management. Steve also served as Editor in Chief of IEEE Software from 1998-2002 and is a member of IEEE Computer Society and ACM. Steve earned a master's degree in software engineering from Seattle University and a bachelor's degree from Whitman College in Walla Walla, Washington

Rebecca Wirfs-Brock, Wirfs-Brock Associates

Rebecca is the inventor of Responsibility-Driven Design and lead author of two books: Object Design: Roles, Responsibilities, and Collaborations (2003), and Designing Object-Oriented Software (1990). She is a board member of the Agile Alliance and an OOPSLA lifer having attended all 20 conferences! Rebecca has been involved with object technology since its infancy. Among her widely used inventions are use case conversations and object role stereotypes. From development on the Tektronix implementation of Smalltalk in the early 1980's, through years of development and training experience, she is recognized as an innovative and influential practitioner of object-oriented design.

A friend recently sent me a link to the World's Smallest Political Quiz. After answering 10 personal value and economic questions, you are rated a Liberal, Centrist, Conservative, Libertarian, or Statist. Depending on my mood I'm a liberal or a centrist. I lean towards supporting personal liberties but believe in some governmental control;especially for the environment. This got me thinking about a similar test for software folks. Do you lean towards agile, centrist, traditional, anarchic, or formal software development practices? If so, what values do you hold? We need to understand what drives process differences as well as what values we hold in common. Agilists and traditionalists both value process. Anarchists don't—whatever works, works. Neither do formalists—they value formalisms over any process. Yet there are differences. Agilists value reactive planning more; traditionalists prefer predictive planning. Yet most want to eliminate defects sooner and could benefit from Test Driven Development. You'll be at the top of your game if you look for practices that support your values regardless of their origin. In this belief I am a centrist and an opportunist.

Ed Yourdon, NODRUOY

Edward Yourdon is an internationally-recognized computer consultant, as well as the author of more than two dozen books, including Byte Wars, Managing High-Intensity Internet Projects, Death March, Rise and Resurrection of the American Programmer, and Decline and Fall of the American Programmer. His latest book, 'Outsource: Competing in the Global Productivity Race', discusses both current and future trends in offshore outsourcing, and provides practical strategies for individuals, small businesses, and the nation to cope with this unstoppable tidal wave. Yourdon is widely known as the lead developer of the structured analysis/design methods of the 1970s, as well as co-developer of the Yourdon/Whitehead method of object-oriented analysis/design and the popular Coad/Yourdon OO methodology of the late 1980s and 1990s. Yourdon has worked in the computer industry for 40 years, beginning when Digital Equipment Corporation hired him in 1964 to write the FORTRAN math library for the PDP-5 and the assembler for the popular PDP-8 minicomputer. During his career, he has worked on over 25 different mainframe computers and was involved in a number of pioneering computer technologies, such as time-sharing operating systems and virtual memory systems. After stints with DEC and GE, a small consulting firm, and a few years as an independent consultant, Yourdon founded his own consulting firm, YOURDON Inc., in 1974, in order to provide educational, publishing, and consulting services in state-of-the-art software engineering technology. Yourdon is the author of over 500 technical articles; he has also written 27 computer-related books since 1967. Among his recent books are 'Death March' (1997), 'Case Studies in Object-Oriented Analysis and Design' (1996), 'Mainstream Objects' (1995), and 'Object-Oriented Systems Development: An Integrated Approach' (1994), as well as two earlier OO books co-authored with Peter Coad. In addition to serving on the Board of Directors of iGate Capital Corp, and India-based Mascot Systems Corp, Yourdon also served on the Defense Department's Airlie Council, an advisory group that focused on finding "best practice" guidelines and techniques for large, complex projects throughout the 1990s. In addition, he serves on the Technical Advisory Board of two high-tech companies: Interelate Inc., and Aspen Technology Inc. Yourdon was an advisor to Technology Transfer's research project on software industry opportunities in the former Soviet Union, and a member of the expert advisory panel on I-CASE acquisition for the U.S. Department of Defense. He is also the Director of the Business/IT Trends Service for the Cutter Consortium, of which he is co-founder and chairman, and he serves as Editor Emeritus of the Consortium's flagship publication, the Cutter IT Journal. Ed Yourdon received a B.S. in Applied Mathematics from MIT in 1965; he has carried out graduate work at MIT and at the Polytechnic Institute of New York. He has been appointed an Honorary Professor of Information Technology at Universidad CAECE in Buenos Aires, and has lectured at MIT, Harvard, UCLA, and Berkeley.

Living With Legacy: Love It or Leave It?

Thursday, 12:00-13:30 ***Golden West Room***

ABSTRACT: Living With Legacy: Love It or Leave It? As the volume of legacy software grows, how have we grown in our ability to leverage this legacy or, for that matter, is it worth the effort? Is legacy software a hoard of useful information and behavior or is it a ball and chain, something you should cut loose if you want to make progress? Legacy constraints often seem immense and burdensome but, do they always need to be? Is object-oriented legacy software spaghetti code or is it more like ravioli? Do agile methods embrace or reject the use of the legacy?

This panel will address these questions, identify issues and discuss key advances related to evolving legacy software.

CHAIR: Steve Berczuk, Iron Mountain

Steve Berczuk is a consultant, software developer and author. He has been involved in OO development since 1989 and is on staff at Iron Mountain working on their digital archives application. Steve is a member of the ACM, the IEEE, and holds a SB in EE from MIT and a MS in Operations Research from Stanford University.

PANELISTS:

Steven Fraser, QUALCOMM

Steven Fraser is a senior member of staff at QUALCOMM's Learning Center in San Diego, California. From 2003 to 2004 he was an independent consultant in Santa Clara, California. Previous to 2002, Steven held a variety of software technology program management roles at Nortel Networks including: Process Architect, Senior Manager (Disruptive Technology and Global External Research), and Process Engineering Advisor. In 1994 he spent a year as a Visiting Scientist at the Software Engineering Institute (SEI) collaborating with the Application of Software Models project on the development of team-based domain analysis techniques. Steven holds a PhD in EE from McGill University in Montreal and is an avid operatunist and videographer.

Bill Opdyke, North Central College

Bill Opdyke is a faculty member at North Central College in Naperville, IL. Previously, Bill worked at Bell Labs (AT&T and Lucent) as a business/ requirements analyst, software and system architect, applied researcher, product evolution planner, project manager, and corporate trainer. Bill was a panelist for an OOPSLA 97 panel on Discovery Costs.

Michael Feathers, Object Mentor, Inc

Michael Feathers works for Object Mentor, Inc. He is the author of the book Working Effectively with Legacy Code. Michael currently provides training and mentoring in Test-Driven Development (TDD), Refactoring, OO Design, Java, C#, C++, and Extreme Programming (XP). Michael is the original author of CppUnit, a C++ port of the JUnit testing framework, and FitCpp, a C++ port of the FIT integrated-testing framework.

Dennis Mancl, Lucent Technologies

Dennis Mancl is an advocate for use cases, legacy code refactoring, and agile processes at Lucent Technologies - Bell Labs in Murray Hill, NJ.

I'm not sure if I love legacy code, but I've learned to live with it. I think I have been improving my ability to reuse and extend legacy code. Things are improved on the technical side: increased awareness of the importance of architecture, better refactoring tools, use of patterns in the design documentation, and use of scenarios in requirements and design. But management and customers still have unrealistic expectations of legacy-based development. I've learned to watch out for significant changes to the business model. A big change may make the existing system ill-suited to the current problem, so rewriting might be more effective than reusing.

Yoshimi Battles the Pink Robots
[Programmers v. Users: The Culture War]

Thursday, 13:30-15:00 ***Town & Country Room***

ABSTRACT: Who came first --- the user or the programmer? Without programmers, there would be no programs for people to use. Without users, there wouldn't be anyone to pay programmers to write their programs.

In the 1980s, Alan Kay's dream was that object-orientation could bridge the gap between programmers and users. Programs would simulate the world they were suppose to model. Doing with Images would make Symbols. Programming would become the recreation of choice for children of all ages.

But in 2005, the gap between users and programmers is as large as ever. We've got our Dynabooks, but we don't want to be programmers: we just want to play Grand Theft Auto. The manifestos for the latest fads in programing—XML, Agile Development, and Advanced Separation of Concerns—make absolutely no mention of users, just programmers and the business people paying for the software. Meanwhile enrollment in Western computer science and software engineering courses continues to decline.

What does it mean to be a user or a programmer now that all our lives are shaped by software?

PANELISTS:

Jeff Patton, ThoughtWorks Inc.

Jeff Patton has designed and developed software for the past 10 years on a wide variety of projects from on-line aircraft parts ordering to electronic medical records. Since working on an XP team in 2000, Jeff has been heavily involved in agile methods. In particular Jeff has focused on the application of user centered design techniques on agile projects resulting leaner more collaborative forms of traditional UCD practices. Jeff has found that adding UCD thinking to agile approaches of incremental development and story card writing not only makes those tasks easier but results in much higher quality software. Some of his recent writing on the subject can be found at www.abstractics.com/papers and in Alistair Cockburn's Crystal Clear. Jeff's currently a proud employee of ThoughtWorks, an active board member of the Agile Alliance, and founder and list moderator of the agile-usability discussion group on Yahoo Groups.

Brian Foote, University of Illinois

Brian Foote has been programming professionally since the dawn of the Carter Administration, mostly in the service of academic researchers of various stripes. The idiosyncratic demands of these kinds of users played a major role in shaping his perspectives on software design. A consequence of his protracted association with this community was a chronic case of the Stockholm Syndrome, which resulted in his occasional forays into research areas as varied as object-oriented programming, design, reflection, FORTRAN, the Grateful Dead, software architecture, patterns, Photran, Smalltalk, CLOS, metalevel architecture, circumlocution, and software devolution.

Mary Beth Rosson, Professor, Pennsylvania State University

Mary Beth Rosson is Professor of Information Sciences and Technology at The Pennsylvania State University. She received a PhD in experimental psychology in 1982 from the University of Texas. Prior to joining the new School of Information Sciences and Technology at Penn State in 2003, she was a professor of computer science at Virginia Tech for 10 years and a research staff member and manager at IBM's T. J. Watson Research Center for 11 years. Rosson was among the earliest researchers to study the psychological issues associated with the object-oriented paradigm, and spent many years developing and evaluating object-oriented tools and training for professional programmers. One of her abiding interests has been the interplay between the concerns of human-computer interaction and software engineering. Recently she has been studying the tools and practices of end-user developers in educational and general business contexts. Rosson has participated in the OOPSLA conference in many capacities, as a member of the technical program committee as well as the conference committee; she introduced the OOPSLA Educators' Symposium in 1992, the Doctoral Consortium in 1994, and served as General Chair for OOPSLA 2000. She is also active in the ACM SIGCHI community, where she is General Chair for CHI 2007. Rosson is author of Usability Engineering: Scenario-Based Development of Human-Computer Interaction (Morgan Kaufmann, 2002) as well as numerous articles, book chapters, and tutorials. More information is available at http://ist.psu.edu/rosson.

Evan Adams, Google, Inc.

Evan Adams is a software developer at Google. He has been developing software for over two decades. Evan has recently focused on user experience. He has observed that, given the current state of the industry, expecting software developers to produce highly-usable software is absurd; somewhat like expecting the cat to do the laundry. Usability has become a key differentiator in the marketplace. Highly usable software will survive, clumsy software will not. Topics like object-oriented programming and agile methodologies have their place in software development but do not address the fundamental issues necessary to produce highly usable software.

Posters

Chair: Dirk Siebert

The OOPSLA Posters provide an excellent forum for authors to present their work in an informal and interactive setting. Posters are ideal to showcase speculative, late-breaking results or to introduce interesting, innovative work. They allow authors and interested participants to connect to each other and to engage in discussions about the work presented. Posters provide authors with a unique opportunity to draw attention to their work during the conference.

On Display	**Monday, 17:30-19:30**	***Terrace Pavilion and Poolside***
On Display	**Tuesday, 10:00-17:00**	***Terrace Pavilion***
On Display	**Wednesday, 10:00-17:00**	***Terrace Pavilion***
On Display	**Thursday, 10:00-14:00**	***Terrace Pavilion***

Service-oriented Architecture: Third International Workshop Results
Ali Arsanjani, *IBM*
Kerrie Holley, *IBM*

Agile Environments... For the rest of us
Dean Mackie, *Ontario Teachers Pension Plan*
Gifford Louie, *Ontario Teachers Pension Plan*
Jason Rogers, *Ontario Teachers Pension Plan*
Niall Shaw, *Ontario Teachers Pension Plan*

A Technique for Utilizing Optimization Potential During Multicode Identification
Ben Stephenson, *University of Western Ontario*
Wade Holst, *University of Western Ontario*

Self-Designing Software
Mauro Marinilli, *University of Rome*

Predictability by Construction
Paulo Merson, *Software Engineering Institute*
Scott Hissam, *Software Engineering Institute*

Green: A customizable UML class diagram plug-in for Eclipse
Carl Alphonce, *University at Buffalo*
Blake Martin, *University at Buffalo*

YARV: Yet Another RubyVM
Koichi Sasada, *Graduate School of Technology, Tokyo University of Agriculture and Technology*

Universal Geometric Properties of Software Design
Lauren Truesdell, *Inovis*

Beyond the language workbench: a runtime platform for practical semantic computing
Roly Perera, *Dynamic Aspects*
Russ Freeman, *Dynamic Aspects*

Workshop on Domain-Specific Modeling
Juha-Pekka Tolvanen, *MetaCase*
Jonathan Sprinkle, *University of California, Berkeley*
Matti Rossi, *Helsinki School of Economics*

Language Concepts for Improving Reusability in Object-Oriented Software
Marko van Dooren, *University of Leuven*

BoBs: Breakable Objects
Vikram Jamwal, *IIT Bombay*
Sridhar Iyer, *IIT Bombay*

Meta: A Universal Meta-Language for Augmenting and Unifying Language Families, featuring Meta(Oopl) for Object-Oriented Programming Languages
Wade Holst, *The University of Western Ontario*

CodeQuest with DataLog
Elnar Hajiyev, *Programming Tools Group, Oxford University Computing Laboratory*
Mathieu Verbaere, *Programming Tools Group, Oxford University Computing Laboratory*
Oege De Moor, *Programming Tools Group, Oxford University Computing Laboratory*
Kris De Volder, *Software Practices Lab, University of British Columbia*

abc: The AspectBench Compiler for AspectJ (A WorkBench for Aspect-Oriented Programming Language and Compilers Research)
Chris Allan, *Programming Tools Group, Oxford University Computing Laboratory*
Pavel Avgustinov, *Programming Tools Group, Oxford University Computing Laboratory*
Aske Simon Christensen, *BRICS, University of Aarhus*
Bruno Dufour, *Sable Research Group, McGill University, Montreal*
Christopher Goard, *Sable Research Group, McGill University, Montreal*
Laurie Hendren, *Sable Research Group, McGill University, Montreal*
Sascha Kuzins, *Oxford University, United Kingdom*
Jennifer Lhoták, *McGill University, Canada*
Ondrej Lhoták, *McGill University, Canada*
Oege de Moor, *Oxford University, United Kingdom*
Damien Sereni, *Oxford University, United Kingdom*
Ganesh Sittampalam, *Oxford University, United Kingdom*
Julian Tibble, *Oxford University, United Kingdom*
Clark Verbrugge, *McGill University, Canada*

A Model-driven Approach to Formal Refactoring
Tiago Massoni, *Federal University of Pernambuco*
Rohit Gheyi, *Federal University of Pernambuco*
Paulo Borba, *Federal University of Pernambuco*

SelfSync: A Dynamic Round-Trip Environment Engineering
Ellen Van Paesschen, *Vrije Universiteit Brussel*
Wolfgang De Meuter, *Universite des Sciences et Technologies de Lille*
Maja D'Hondt, *Universite des Sciences et Technologies de Lille*

A Comprehensive Model Transformation Approach to Automated Model Construction and Evolution
Yuehua Lin, *University of Alabama at Birmingham*
Jeff Gray, *University of Alabama at Birmingham*

The Future of Data-Centric and Information-Centric Computing
Steve Lakin, *Coventry University*
Sarah Mount, *Coventry University*
Elena Gaura, *Coventry University*
Bob Newman, *Coventry University*
Yvonne Coady, *University of Victoria*
Jeff Eastman, *Windward Solutions*

Posters

Assisting Aspect-Oriented Framework Instantiation: Towards Modeling, Transformation and Tool Support

Marcilio Mendonca, *University of Waterloo*
Paulo Alencar, *University of Waterloo*
Toacy Oliveira, *PUC-RS*
Donald Cowan, *University of Waterloo*

Can Infopipes Facilitate Reuse in a Traffic Application?

Emerson Murphy-Hill, *Portland State University*
Chuan-kai Lin, *Portland State University*
Andrew Black, *Portland State University*
Jonathan Walpole, *Portland State University*

Modeling Turnpike: a Model-Driven Framework for Domain-Specific Software Development

Hiroshi Wada, *Department of Computer Science, University of Massachusetts, Boston*
Junichi Suzuki, *Department of Computer Science, University of Massachusetts, Boston*

Developing Business Object Models with Patterns and Ontologies

Haitham Hamza, *University of Nebraska-Lincoln*

The Squawk Java Virtual Machine: Java on the Bare Metal

Doug Simon, *Sun Microsystems Laboratories*
Cristina Cifuentes, *Sun Microsystems Laboratories*

PAD: A Pattern-Driven Analysis and Design Method

Haitham Hamza, *University of Nebraska-Lincoln*
Yi Chen, *Microsoft Corporation*

High-Level Declarative User Interfaces

Sofie Goderis, *Vrije Universiteit Brussel*
Theo D'Hondt, *Vrije Universiteit Brussel*

Not So eXtreme Programming: Agile Practices for R&D projects

Roberta Coelho, *Pontificia Universidade do Rio de Janeiro (PUC-Rio)*
Esther Brasileiro, *Laboratorio de Sistemas Distribuidos da Universidade de Campina Grande*
Arndt Staa, *Pontificia Universidade do Rio de Janeiro (PUC-Rio)*

Improving Architecture Testability with Patterns

Roberta Coelho, *Pontificia Universidade Catolica do Rio de Janeiro (PUC-Rio)*
Uir'a Kulesza, *Pontificia Universidade Catolica do Rio de Janeiro (PUC-Rio)*
Arndt Staa, *Pontificia Universidade Catolica do Rio de Janeiro (PUC-Rio)*

MVCDC 2: Managing Variabilities consistently in Design and Code

Christa Schwanninger, *Siemens AG*
Danilo Beuche, *pure-systems*
Krzysztof Czarnecki, *University of Waterloo*
Mira Mezini, *Darmstadt University of Technology*
Markus Voelter, *ingenieurbüro für softwaretechnologie*
Rainer Burgstaller, *Siemens AG*

Model Checking the Behavior of Frameworks Extended with Other Frameworks

Federico Balaguer, *University of Illinois at Urbana-Champaign*

Refactoring the JUnit Framework using Aspect-Oriented Programming

Uira Kulesza, *Pontificia Universidade Catolica of Rio de Janeiro*
Claudio Sant'Anna, *Pontificia Universidade Catolica of Rio de Janeiro*
Carlos Lucena, *Pontificia Universidade Catolica of Rio de Janeiro*

Security Pattern and Evolution of MTA Architecture

Munawar Hafiz, *University of Illinois, Urbana-Champaign*

Implementing Incrementalization Across Object Abstraction

Michael Gorbovitski, *Computer Science Dept., State Univ. of New York at Stony Brook*
Tom Rothamel, *Computer Science Dept., State Univ. of New York at Stony Brook*
Yanhong Liu, *Computer Science Dept., State Univ. of New York at Stony Brook*
Scott Stoller, *Computer Science Dept., State Univ. of New York at Stony Brook*

SDD: High performance code clone detection system for large scale source code

Seunghak Lee, *Dept. of Chemistry, POSTECH*
Iryoung Jeong, *Depf. of Computer Science Education, Korea University*

Ambient-Oriented Programming in AmbientTalk

Stijn Mostinckx, *Vrije Universiteit Brussel*
Tom Van Cutsem, *Vrije Universiteit Brussel*
Jessie Dedecker, *Vrije Universiteit Brussel*
Sebastián Gonzalez, *Université catholique de Louvain*
Wolfgang De Meuter, *Vrije Universiteit Brussel*
Theo D'Hondt, *Vrije Universiteit Brussel*

Model-Driven Software Product Lines

Krzysztof Czarnecki, *University of Waterloo*
Michal Antkiewicz, *University of Waterloo*

Scopira: An Open Source C++ Framework for Biomedical Data Analysis Applications—A Research Project Report

Aleksander Demko, *University of Manitoba, Institute of Biodiagnostics*
Rodrigo Vivanco, *Institute of Biodiagnostics*
Nick Pizzi, *Institute of Biodiagnostics*

Service Orchestration Patterns: Graduating from State of the Practice to State of the Art

Dragos Manolescu, *ThoughtWorks, Inc.*
Boris Lublinsky, *CNA Insurance*

Fourth Killer Examples for Design Patterns and Objects First Workshop Results

Carl Alphonce, *University at Buffalo*
Stephen Wong, *Rice University*
Michael Caspersen, *University of Aarhus*
Adrienne Decker, *University at Buffalo*

Visualizing Errors in Object Oriented Programs

Hani Girgis, *University at Buffalo, The State University of New York*
Bharat Jayaraman, *University at Buffalo, The State University of New York*
Paul Gestwicki, *University at Buffalo, The State University of New York*

Supporting Configuration and Deployment of Component-based DRE Systems Using Frameworks, Models, and Aspects

Gan Deng, *Vanderbilt University*

Incremental Exploratory Visualization of Relationships in Large Codebases for Program Comprehension

Vineet Sinha, *MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)*
Rob Miller, *MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)*
David Karger, *MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)*

Chair: Robert Biddle, Carleton University, Ottawa, Canada

Practitioner Reports are an integral part of the OOPSLA technical program. These reports provides a large group of peers the opportunity to learn from a project's experience applying software technologies or related practices. For many OOPSLA attendees, these reports are the most important part of the conference.

Practitioner reports explore how concepts that sound good on paper (and at conferences!) work on real projects. They are a valuable means of communicating experiences, especially at the "bleeding edge". Many attendees want to find out what it is like to adopt a new language, use new engineering methods, integrate commercial software, develop web services applications, develop frameworks, use design patterns, etc. Expectations, beliefs, and hopes can be validated, or dashed, by the experience that is reported.

OOPSLA practitioner reports present experience and reflections, together with supporting evidence for any claims made. And they particularly include reports that discuss both benefits and drawbacks of the approaches used. Reports may focus on a particular aspect of technology usage and practice, or describe broad project experiences. Some reports also focus on people, process or development challenges.

Practice and Principles — Tuesday, 15:30-17:00 — *Golden West Room*

Session Chair: Robert Biddle, Carleton University

Legacy System Exorcism by Pareto's Principle

Kristoffer Kvam, Telenor Nordic, IT, CRM

Daniel Bakkelund, Telenor Nordic, IT, CRM

Rodin Lie, Telenor Nordic, IT, CRM

Exorcism is mainly thought of as the rite of driving out the Devil and his demons from possessed persons. This text is about the same process except here the target is a legacy software system. The target system was a major component based system having been developed over 7 years by 30 to 60 people continuously under a classic plan driven approach. The Pareto Principle, or 80/20 rule as it is often called, is used as the framework to prioritize activities in a major reengineering initiative on the system from limited resources. The initiative's main focus was to increase the developer productivity in the maintenance project in the system by 25 percent. Typical agile practices were the inspiration for many of the changes implemented through the project.

A measurement program is presented for validating success, and the XRadar open source tool is used for measuring the program. In one year, the productivity increase was above 30 percent. There seems to be a high correlation between productivity and the implementation of the agile practices such as short iterations, daily standup-meetings and pair programming as substitutes with the practice of a formal QA regime. During the same period the error proneness of the system decreased with several magnitudes and our definition of the internal software quality increased by 22 percent. Hence, based on our measurements, the increased productivity was not substituted by lower quality in the system - on the contrary.

Estimating Software Based on Use Case Points

Edward Carroll, Agilis Solutions, a business unit of Hepieric, Inc.

It is well documented that software product cost estimates are notoriously inaccurate across the software industry. Creating accurate cost estimates for software product development projects early in the product development lifecycle has always been a challenge for the industry. This article describes how a large multi-team software engineering organization (over 450 engineers) estimates project cost accurately and early in the software development lifecycle using Use Case Points, and the process of evaluating metrics to ensure the accuracy of the model.

The engineering teams of Agilis Solutions in partnership with FPT Software, provide our customers with accurate estimates for software product projects early in the product lifecycle. The basis for these estimates are initial definitions of Use Cases, given point factors and modified for technical and environmental factors according to the Use Case Point method defined within the Rational Unified Process. After applying the process across hundreds of sizable (60 man-months average) software projects, we have demonstrated metrics that prove an estimating accuracy of less than 9% deviation from actual to estimated cost on 95% of our projects. Our process and this success factor is documented over a period of five years, and across more than 200 projects.

Finding the Forest in the Trees

Jeff Patton, ThoughtWorks Inc.

While the iterative development approaches found in Agile Software Development fulfill the promise of working software each iteration, that task of choosing which software to build first can be daunting. While simple guidelines like "choose features with the highest business value" may seem useful, what really has high business value may be debatable. On large projects, the number of possible features to implement may number in the hundreds. Prioritizing such a list can be exhausting and frustrating. Simply building the features that are highest priority on such a list often results in a software that's unusable by its intended audience because of omitted features that, although lower priority, were necessary to the users daily workflow. Wading through piles of features to create a successful project release plan requires strategies other than "highest business value". We need strategies that bring our focus above the feature level, the tree level, to see the forest. Strategies that help us identify small but coherent sets of functionality that by releasing them early will begin to really generate the business value we desire.

This experience report will discuss my team's experience working within large healthcare company writing software for use in their hospitals newborn intensive care unit. The very large scope of this project and the urgent need for delivery made project release planning difficult. Focusing on feature details, user story writing, lead to more confusion about priorities and release strategy. Making good use of User Centered Design user role models and task models gave us the big picture we need to un-stick the release planning process and effectively choose the bit of project scope we need to focus on for our first and subsequent releases.

Keywords: Agile, User Centered Design, Incremental Release Planning, User Story Writing

Language and Reality — Wednesday, 10:30-12:00 — *Golden West Room*

Session Chair: Alain Désilets, National Research Council, Canada

Honey, I shrunk the Types—How Behavioral Types lose relevance on the edges of OO Applications

John Kuriakose, Infosys Technologies

How Behavioral Types lose relevance on the edges of OO Applications and why a core Data fabric is useful for adaptability?

OO Programs are built by first defining User Types within the language environment and then realizing program requirements by using the behavior defined by these Types.

We argue against defining types to deal with every scenario. OO Programs within an enterprise have to deal with the non-OO world that includes RDBMS, Other Applications, and Humans etc. On these EDGES that OO Programs interact with the non-OO world we have observed that there is little respect for Types and behavior and the requirements and expectation is most often data. If this is true, then OO application developers will repeatedly have to extract Data to and from Objects defined by Types to support the pure data interface that the non OO world supports.

In this paper we attempt to highlight some benefits we have realized by equipping an OO Application (Banking Middleware in Java) with a core representation for data which we refer to as the "data fabric". We further show that seemingly unrelated problems in Database persistence, data-binding, message data transformation and metadata management that appear within an Enterprise context can now be addressed well within the data fabric. The Data fabric is itself realized using reflective uniform data API defined using a few Types that abstract data definition as Meta-data.

This then reopens the case for OO developers to enhance their effectiveness with generic Data Models and principles of Data Driven Programming.

Removing duplication from java.io: a case study using Traits

Emerson Murphy-Hill, Portland State University **Andrew Black,** Portland State University

Philip Quitslund, Portland State University

Code duplication is a serious problem with no easy solution, even in industrial-strength code. Single inheritance cannot provide for effective code reuse in all situations, and sometimes programmers are driven to duplicate code using copy and paste. A language feature called traits enables code to be shared across the inheritance hierarchy, and claims to permit the removal of most duplication. We attempted to validate this claim in a case study of the java.io library. Detecting duplication was more complex than we had imagined, but traits were indeed able to remove all that we found.

Arithmetic with Measurements on Dynamically Typed Object Oriented Languages

Hernan Wilkinson, Mercap Software Corp. **Luciano Romeo,** Mercap Software Corp.

Maximo Prieto, Lifia, Universidad Nacional de La Plata

In physics, like in other sciences, formulas are specified using explicit measurements, that is, a number with its unit. The first step to determine the validity of a physics formula's evaluation is to verify that the unit of the result corresponds with the prospective unit. In software development, physics, financial and other sciences formulas are programmed using mathematical expressions based only on numbers, being the units of these numbers implicitly given by the semantics of the program or assumed by the programmer's knowledge. Consequently, it is common that errors result from operating with values expressed in different units, e.g., dividing a quantity of years by a quantity of months, without obtaining any type of indication or objection to this error from the system. In this report we discuss our experience designing and implementing a model that solves this problem reifying the concept of measurement, unit and their arithmetic. Our model relieves the programmer from the arduous task of verifying the validity of the arithmetic expressions regarding units, delegating that responsibility to the system, thereby diminishing the errors introduced by the incorrect use of values expressed in different units. We also show that having implemented this model with a dynamically typed language simplified its programming and increased its reusability.

Plans and Results **Thursday, 13:30-15:00** ***Golden West Room***

Session Chair: Robert Biddle, Carleton University

Service-Oriented Architecture and Business Process Choreography in an Order Management Scenario: Rationale, Concepts, Lessons Learned

Olaf Zimmermann, IBM Global Services
Jonas Grundler, IBM Software Group
Vadim Doubrovski, IBM Global Services
Kerard Hogg, IBM Global Services

Effective and affordable business-to-business process integration is a key success factor in the telecommunications industry. A large telecommunication wholesaler, supplying its services to more than 150 different service retailers, enhanced the process integration capabilities of its core order management system, consisting of more than 100 complex functions, through widespread use of SOA, business process choreography and Web services concepts.

On this project, challenging requirements such as complexity of business process models and multi-channel accessibility turned out to be true proof points for the applied SOA concepts, tools, and runtime environments. To implement an automated and secure Business-to-Business (B2B) channel, to achieve interoperability between script-based clients and a Java middle tier, and to introduce a process choreography layer into a large existing application were some of the key requirements that had to be addressed. The solution complies with the Web Services Interoperability Basic Profile 1.0, and makes use of executable business process models defined in the Business Process Execution Language for Web Services (BPEL).

This paper discusses the rationale behind the decision for SOA, process choreography, and Web services, and gives an overview of the BPEL-centric process choreography architecture. Furthermore, it features lessons learned and best practices identified during design, implementation and rollout of the solution.

Agility vs. Stability at a Successful Startup

Kurt Madsen, MetaTech, Inc.

It is not uncommon for good technical solutions to fail in the marketplace. Equally true, great business opportunities are not always met with appropriate technical solutions. This paper presents lessons learned from developing a communications framework in a thriving start-up. In particular, the challenges of meeting and adapting to evolving business models and product definition are presented.

Using Predicate Fields in a Highly Flexible Industrial Control System

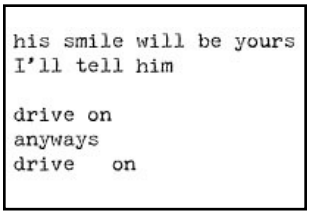
Shay Artzi, MIT
Michael Ernst, MIT

Predicate fields allow an object's structure to vary at runtime based on the object's state: a predicate field is present or not, depending on the values of other fields. Predicate fields and related concepts have not previously been evaluated outside a research environment. We present a case study of two industrial applications with similar requirements, one of which uses predicate fields and one of which does not. The use of predicate fields was motivated by requirements for high flexibility, by unavailability of many requirements, and by high user interface development costs.

Despite an implementation of predicate fields as a library (rather than as a language extension), developers found them natural to use, and in many cases they significantly reduced development effort.

Research Papers — Tuesday Morning

Chair: Richard P. Gabriel, Sun Microsystems, Inc.



Research papers form the heart of the OOPSLA experience. Research papers describe substantiated new research or novel technical results, advance the state of the art, or report on significant experience or experimentation. The 29 technical papers were selected after a rigorous peer review of 142 submissions by an international program committee consisting of 28 experts representing the spectrum of object technology. Each paper was assigned to at least three reviewers with many reviewed by more, and one by nine. The committee met face-to-face for two days in Los Angeles, California. The papers selected for presentation at this conference should advance the state of the art of object technology in significant ways.

Type Types — Tuesday, 10:30-12:00 — *San Diego Room*

Session Chair: Michael Kölling, University of Kent

Associated Types and Constraint Propagation for Mainstream Object-Oriented Generics

Jaakko Järvi, Texas A&M University
Andrew Lumsdaine, Indiana University
Jeremiah Willcock, Indiana University

Support for object-oriented programming has become an integral part of mainstream languages, and more recently generic programming has gained widespread acceptance as well. A natural question is how these two paradigms, and their underlying language mechanisms, should interact. One particular design option, that of using subtyping to constrain the type parameters of generic functions, has been chosen for the generics extensions to Java and C#. Certain shortcomings have previously been identified in using subtyping for constraining parametric polymorphism in the context of generic programming. To address these, we propose the expansion of object-oriented interfaces and subtyping to include associated types and constraint propagation. Associated types are type members of interfaces and classes. Constraint propagation allows certain constraints on type parameters to be inferred from other constraints on those parameters and their use in base class type expressions. The paper demonstrates these extensions in the context of Generic C# and presents a formalism proving their safety. The formalism is applicable to other mainstream OO languages supporting F-bounded polymorphism, such as Java.

Generalized Algebraic Data Types and Object-Oriented Programming

Andrew Kennedy, Microsoft Research Ltd
Claudio Russo, Microsoft Research Ltd

Generalized algebraic data types (GADTs) have received much attention recently in the functional programming community. They generalize the type-parameterized datatypes of ML and Haskell by permitting constructors to produce different type-instantiations of the same datatype. GADTs have a number of applications, including strongly typed evaluators, generic pretty-printing, generic traversals and queries, and typed LR parsing. We show that existing object-oriented programming languages such as Java and C# can express GADT definitions, and a large class of GADT-manipulating programs, through the use of generics, subclassing, and virtual dispatch. However, some programs can be written only through the use of redundant run-time casts. We propose a generalization of the type constraint mechanisms of C# and Java to avoid the need for such casts, present a Visitor pattern for GADTs, and describe a switch construct as an alternative to virtual dispatch on datatypes. We formalize both extensions and prove a type soundness result.

Keywords: generalised algebraic data types, patterns, generics, C#, Java, virtual methods, constraints, polymorphism, existential types

Scalable Component Abstractions

Martin Odersky, EPFL
Matthias Zenger, Google

We identify three programming language abstractions for the construction of reusable components: abstract type members, explicit self-types, and symmetric mixin composition. Together, these abstractions enable us to transform an arbitrary assembly of static program parts with hard references between them into a system of reusable components. The transformation maintains the structure of the original system. We demonstrate this approach in two case studies, a subject/observer framework and a compiler front-end.

Analysis Analyzed **Tuesday, 13:30-15:00** ***San Diego Room***

Session Chair: Dave West, New Mexico Highlands

Demand-Driven Points-to Analysis for Java

Manu Sridharan, UC Berkeley
Lexin Shan, UC Berkeley
Denis Gopan, University of Wisconsin
Rastislav Bodik, UC Berkeley

We develop pointer analyses suitable for use highly time-constrained environments, such as just-in-time (JIT) compilers and interactive development environments (IDEs). In such environments, the running time of the analysis is critical, leading one to consider demand-driven solutions. In the worst case, however, a demand-driven analysis can take as long as an exhaustive analysis. For long-running queries, such environments could use early termination and simply stop the analysis after some timeout. Previous demand-driven pointer analyses do not perform well under such early termination constraints.

We formulate Andersen's analysis for Java as a CFL-reachability problem, and observe that it has the balanced-parentheses property seen in other program analysis problems. We use this observation to develop two new demand-driven pointer analysis algorithms, DemandFB and TargetedFS, that are suitable for use with early termination. DemandFB is an extremely simple field-based algorithm that essentially performs a depth-first search of the CFL-reachability graph. TargetedFS iteratively adds field-sensitivity to DemandFB where profitable, exploiting the balanced-parentheses structure for efficiency. We show that DemandFB is a very effective algorithm under early termination; for our clients, it can answer 89-94% of virtual call queries as precisely as an exhaustive field-sensitive Andersen's analysis, with each query taking less than 2ms. DemandFB can answer all queries in hot methods of javac with the same precision as an exhaustive field-sensitive Andersen's analysis, with a 34x speedup (and a 16x speedup over exhaustive field-based analysis). We also show that given a limited budget, TargetedFS is better than known demand-driven pointer analyses at closing the gap between DemandFB and exhaustive field-sensitive Andersen's.

Deriving Object Typestates in the Presence of Inter-Object References

Mangala Gowri Nanda, IBM India Research Lab
Satish Chandra, IBM India Research Lab
Christian Grothoff, Purdue University

We are interested in static analysis of Java classes with the goal of discovering the preconditions under which a certain program point within a method may be reached, taking into account the effects of previous method calls on an object of that class. The information pertinent for this computation is represented as the object's typestate, which is a finite set of relevant predicates that abstract the object's actual state. The execution of a method depends on an object's current typestate as well as other input parameters; the object may transition to a different typestate during the method's execution.

It is common for objects to contain references to other objects. In such cases, an object's behavior may depend on, in addition to its own state, the state of objects it has a reference to. The main contribution of this paper is to discover relevant object typestates, as well as transitions between typestates, in the presence of inter-object references. Our analysis first performs a combined predicate discovery and predicate abstraction to derive boolean versions of Java classes given as input. It then uses abstract interpretation to compute the typestate transitions caused by method calls. A novel aspect of this work is that a set of Java classes is analyzed in isolation, without any client program being provided. To do this, the analysis simulates all possible client's actions via a synthetic heap, all of whose interesting configurations are explored by our analysis.

The information we compute can be put to use in several ways. It can be used in checking whether a given client code erroneously uses a set of Java classes in a way that can throw an exception. It can also be used in creating test drivers for Java classes in order to exercise all relevant code paths in the corresponding methods.

Micro Patterns in Java Code

Joseph (Yossi) Gil, Technion-Israel Institute of Technology
Itay Maman, Technion-Israel Institute of Technology

Micro patterns are similar to design patterns, except they are mechanically recognizable and stand at a lower, closer to the implementation, level of abstraction. Each pattern can be written as a simple formal condition on the class structure.This paper presents a catalog of 27 micro-patterns defined on Java on Java classes and interfaces. The catalog captures a wide spectrum of common programming practices,including a particular and (intentionally restricted) use of inheritance, immutability, data management and wrapping,restricted creation, and emulation of procedural-, modular- and even functional-programming paradigms in object oriented programming. Together, the patterns present a set of prototypes after which about 75% of all Java classes and interfaces are modeled.

A statistical analysis of occurrences of micro patterns ina large data set, spanning some 70,000 Java classes drawn from a rich set of application domains, shows, with high confidence level that the use of these patterns reflects consciousness and discernible design decisions, which is sustained in the software evolution. With high confidence level, we can also show that the use of these patterns is tied to the specification, or the purpose, that the software realizes. The traceability, abundance and the statistical significance of micropattern occurrence raises the hope of using the classification of software into these patterns for a more founded appreciation of its design and code quality.

Archetypal Architectures **Tuesday, 15:30-17:30** ***San Diego Room***

Session Chair: Karl Lieberherr, Northeastern

ArchMatE: From Architectural Styles to Object-Oriented Models through Exploratory Tool Support

J. Andrés Díaz Pace, ISISTAN Research Institute, Faculty of Sciences, UNICEN University

Marcelo R. Campo, ISISTAN Research Institute, Faculty of Sciences, UNICEN University

Given the difficulties of conventional object technologies to deal with quality-attribute concerns, software architectures appear as an interesting approach to manage them better. A problem to make this approach feasible is the gap between architectural and object models. Succeeding in bridging these two worlds implies that those design decisions about quality attributes made at the architectural level should be reflected and at the object level. Nonetheless, there exist usually multiple, different materializations for a given architecture. Furthermore, any materialization requires considerable design background and experience from the developer. In this paper, we describe a tool approach, called ArchMatE, to assist developers in the exploration of object-oriented solutions for grounding architectural models. An important aspect of the approach is that the materializations are accomplished by means of quality-oriented strategies, so that those concerns prescribed by the original architecture are mostly preserved.

Modeling Architectural Patterns Using Architectural Primitives

Uwe Zdun, Vienna University of Economics

Paris Avgeriou, Fraunhofer IPSI

Architectural patterns are a key point in architectural documentation. Regrettably, there is poor support for modeling architectural patterns, because they do not directly match the elements in modeling languages, and, at the same time, they support an inherent variability that is hard to model using a single modeling solution. This paper proposes tackling this problem by finding and representing architectural primitives, as the participants in the solution that patterns convey. In particular, we examine a number of architectural patterns to discover those primitive abstractions that are common among the patterns, and at the same time demonstrate a degree of variability in each pattern. These abstractions belong in the components and connectors architectural view. We have selected UML 2 as the language for representing these primitive abstractions as extensions of the standard UML elements. The added value of this approach is twofold: it proposes a generic and extensible approach for modeling architectural patterns by means of architectural primitives; it demonstrates an initial set of primitives that participate in several well-known architectural patterns.

Parametric Polymorphism for Software Component Architectures

Cosmin Oancea, University of Western Ontario

Stephen Watt, University of Western Ontario

Parametric polymorphism has become a common feature of mainstream programming languages, but software component architectures have lagged behind and do not support this feature. We examine the problem of providing parametric polymorphism in this setting.

The details of generics, templates or functors, as they are variously called, differ significantly in different programming languages. We have investigated how to resolve different binding times and parametric polymorphism semantics in a range of representative programming languages, and have identified a common ground that can be suitably mapped to different language bindings. We present a generic component architecture extension that provides support for parameterized components, and can be easily adapted to work on top of various software component architectures in use today (e.g., CORBA, JNI, DCOM). We have implemented and tested our extension on top of CORBA.

We present Generic Interface Description Language (GIDL), an extension to CORBA-IDL, supporting generic types, and our language bindings for C++, Java, and Aldor. We describe our implementation of GIDL, consisting of a GIDL to IDL compiler and tools for generating linkage code under the language bindings. Furthermore, we describe how this architecture can be used to expose C++'s STL and Aldor's BasicMath libraries to a multi-language environment, and discuss our mappings in the context of automatic library interface generation.

Using GIDL, component-based applications can enjoy the accepted benefits of generic programming: provide clearer and more precise specifications, eliminating ambiguities in the object interface definition, and ultimately exhibit a greater degree of component re-use.

KEYWORDS: Parametric polymorphism, generics, software component architecture, multi-language programming, interface definition languages

Using Dependency Models to Manage Complex Software Architecture

Neeraj Sangal, Lattix, Inc.

Vineet Sinha, Massachusetts Institute of Technology

Ev Jordan, Lattix, Inc.

Daniel Jackson, Massachusetts Institute of Technology

This paper describes a new approach, based on the Design Structure Matrix (DSM), which uses inter-module dependencies to specify and manage the architecture of large software systems. The system is decomposed into a hierarchy of subsystems with the dependencies between the subsystems presented in the form of an adjacency matrix. The hierarchic decomposition allows a succinct definition of design rules to specify allowable dependencies.

This yields a number of key benefits: Our adaptation of the DSM to represent dependencies with subsystem hierarchies appears to overcome scaling problems commonly associated with directed graph representations. DSM also provides an intuitive framework to specify and view design rules. It is easy to identify and enforce common architectural patterns such as layering and the use of components. A variety of algorithms are available to help organize the matrix in a form that reflects the architecture and highlights patterns and problematic dependencies.

A tool has been developed and applied to a number of large systems; this paper describes its application to the reengineering of Haystack, an information retrieval system.

Language Lingo **Wednesday, 10:30-12:00** ***San Diego Room***

Session Chair: William Cook, UT, Austin

Classbox/J: Controlling the Scope of Change in Java

Alexandre Bergel, SCG - University of Bern
Oscar Nierstrasz, SCG - University of Bern
Stéphane Ducasse, SCG - University of Bern

Unanticipated changes to complex software systems can introduce anomalies such as duplicated code, suboptimal inheritance relationships and a proliferation of run-time downcasts. Refactoring to eliminate these anomalies may not be an option, at least in certain stages of software evolution.

Classboxes are modules that restrict the visibility of changes to selected clients only, thereby offering more freedom in the way unanticipated changes may be implemented, and thus reducing the need for convoluted design anomalies.In this paper we demonstrate how classboxes can be implemented in statically-typed languages like Java. We also present an extended case study of Swing, a Java GUI package built on top of AWT, and we document the ensuing anomalies that Swing introduces. We show how Classbox/J, a prototype implementation of classboxes for Java, is used to provide a cleaner implementation of Swing using local refinement rather than subclassing.

Interaction-Based Programming with Classages

Yu David Liu, The Johns Hopkins University
Scott Smith, The Johns Hopkins University

This paper presents Classages, a novel class-based, interaction-centric object-oriented language. Classes and objects in Classages are fully encapsulated, with explicit interfaces for all interactions they might be involved in. The design of Classages touches upon a wide range of language design topics, including encapsulation, object relationship representation, and object confinement. An encoding of Java's OO model in Classages is provided, showing how standard paradigms are supported. A prototype Classages compiler is described.

Javari: Adding Reference Immutability to Java

Matthew S. Tschantz, MIT
Michael D. Ernst, MIT

This paper describes a type system that is capable of expressing and enforcing immutability constraints. The specific constraint expressed is that the abstract state of the object to which an immutable reference refers cannot be modified using that reference. The abstract state is (part of) the transitively reachable state: that is, the state of the object and all state reachable from it by following references. The type system permits explicitly excluding fields or objects from the abstract state of an object. For a statically type-safe language, the type system guarantees reference immutability. If the language is extended with immutability downcasts, then run-time checks enforce the reference immutability constraints.

This research builds upon a previous research in language support for reference immutability. Improvements that are new in this paper include distinguishing the notions of assignability and mutability; integration with Java 5's generic types and with multi-dimensional arrays; a mutability polymorphism approach to avoiding code duplication; type-safe support for reflection and serialization; and formal type rules for a core calculus. Furthermore, it retains the valuable features of the previous dialect, including usability by humans (as evidenced by experience with 160,000 lines of Javari code) and interoperability with Java and existing JVMs.

Keywords: type system, verification, immutability, readonly, mutable, Javari, Java, const, generic types

Adaptation Adapted **Wednesday, 13:30-15:00** ***San Diego Room***

Session Chair: Andrew P. Black, Portland State

Fine-Grained Interoperability through Mirrors and Contracts

Kathryn Gray, University of Utah **Matthew Flatt,** University of Utah
Robert Findler, University of Chicago

As a value flows across the boundary between interoperating languages, it must be checked and converted to fit the types and representations of the target language. For simple forms of data, the checks and coercions can be immediate; for higher order data, such as functions and objects, some must be delayed until the value is used in a particular way. Typically, these coercions and checks are implemented by an ad-hoc mixture of wrappers, reflection, and dynamic predicates. We observe that 1) the wrapper and reflection operations fit the profile of mirrors, 2) the checks correspond to contracts, and 3) the timing and shape of mirror operations coincide with the timing and shape of contract operations. Based on these insights, we present a new model of interoperability that builds on the ideas of mirrors and contracts, and we describe an interoperable implementation of Java and Scheme that is guided by the model.

Pluggable AOP: Designing Aspect Mechanisms for Third-party Composition

Sergei Kojarski, Northeastern University **David Lorenz,** Northeastern University

Studies of Aspect-Oriented Programming (AOP) usually focus on a language in which a specific aspect extension is integrated with a base language. Languages specified in this manner have a fixed, non-extensible AOP functionality. In this paper we consider the more general case of integrating a base language with a set of domain specific third-party aspect extensions for that language. We present a general mixin-based method for implementing aspect extensions in such a way that multiple, independently developed, dynamic aspect extensions can be subject to third-party composition and work collaboratively.

Refactoring Support for Class Library Migration

Ittai Balaban, New York University **Robert Fuhrer,** IBM T.J. Watson Research Center
Frank Tip, IBM T.J. Watson Research Center

As object-oriented class libraries evolve, classes are occasionally deprecated in favor of others with roughly the same functionality. In Java's standard libraries, for example, class Hashtable has been superseded by HashMap, and Iterator is now preferred over Enumeration.

Migrating client applications to use the new idioms is often desirable, but making the required changes to declarations and allocation sites can be quite labor-intensive. Moreover, migration becomes complicated—and sometimes impossible—if an application interacts with external components, if a deprecated class is not completely equivalent to its replacement, and if multiple interdependent types must be migrated simultaneously.

We present an approach in which mappings between legacy types and their replacements are specified by the programmer. Then, an analysis based on type constraints determines where declarations and allocation sites can be updated. The method was implemented in Eclipse, and evaluated on a number of Java applications. On average, our tool could migrate more than 90% of the references to legacy types in these benchmarks.

Machine Machinery **Wednesday, 15:30-17:30** ***San Diego Room***

Session Chair: David F. Bacon, IBM

Automating Vertical Profiling

Matthias Hauswirth, University of Colorado
Amer Diwan, University of Colorado
Peter Sweeney, IBM Thomas J. Watson Research Center
Michael Mozer, University of Colorado

Last year at OOPSLA we presented a methodology, "vertical profiling", for understanding the performance of object-oriented programs. The key insight behind this methodology is that modern programs run on top of many layers (virtual machine, middleware, etc) and thus we need to collect and combine information from all layers in order to understand system performance. Although our methodology was able to explain previously unexplained performance phenomena, it was extremely labor intensive. In this paper we describe and evaluate techniques for automating two significant activities of vertical profiling: trace alignment and correlation.

Trace alignment aligns traces obtained from separate runs so that one can reason across the traces. We are not aware of any prior approach that effectively and automatically aligns traces. Correlation sifts through hundreds of metrics to find ones that have a bearing on a performance anomaly of interest. In prior work we found that statistical correlation was only sometimes effective.

We have identified highly-effective automated approaches for both activities. For aligning traces we explore dynamic time warping, and for correlation we explore eight correlators: Pearson's coefficient, Spearman's coefficient, Manhattan distance, Euclidean distance, dynamic time warping, manual linear pattern, best splits, and same splits. Although we explore these activities in the context of vertical profiling, both activities are widely applicable in the performance analysis area.

Improving Virtual Machine Performance Using a Cross-Run Repository

Matthew Arnold, IBM T.J. Watson Research Center
V.T. Rajan, IBM T.J. Watson Research Center
Adam Welc, Purdue University / IBM Research

Virtual machines for languages such as the Java programming language make extensive use of online profiling and dynamic optimization to improve program performance. But despite the important role that profiling plays in achieving high performance, current virtual machines discard a program's profile data at the end of execution, wasting the opportunity to use past knowledge to improve future performance.

In this paper, we present a fully automated architecture for exploiting cross-run profile data in virtual machines. Our work addresses a number of challenges that previously limited the practicality of such an approach.

We apply this architecture to address the problem of selective optimization, and describe our implementation in IBM's J9 Java virtual machine. Our results demonstrate substantial performance improvements on a broad suite of Java programs, with the average performance ranging from 8.8% to 16.6% depending on the execution scenario.

Quantifying the Performance of Garbage Collection vs. Explicit Memory Management

Matthew Hertz, University of Massachusetts Amherst
Emery Berger, University of Massachusetts Amherst

Garbage collection yields numerous software engineering benefits, but its quantitative impact on performance remains elusive. One can measure the cost of conservative garbage collection relative to explicit memory management in C/C++ programs by linking in an appropriate collector. This kind of direct comparison is not possible for languages designed for garbage collection (e.g., Java), because programs in these languages naturally do not contain calls to free. Thus, the actual gap between the time-space performance of explicit memory management and precise, copying garbage collection remains unknown.

We take the first steps towards quantifying the performance of precise garbage collection versus explicit memory management. We present a novel experimental methodology that lets us treat unaltered Java programs as if they used explicit memory management. Our system generates exact object reachability information by processing heap traces with the Merlin algorithm. It then re-executes the program, invoking free on objects just before they become unreachable. Since this point is the latest that a programmer could explicitly free objects, our approach conservatively approximates explicit memory management. By executing inside an architecturally-detailed simulator, this "oracular" memory manager eliminates the effects of trace processing while measuring the costs of calling malloc and free.

We compare explicit memory management to both copying and non-copying garbage collectors across a range of benchmarks, and include non-simulated runs that validate our results. Our results quantify the time-space tradeoff of garbage collection: with five times as much memory, the Appel-style generational garbage collector matches the performance of explicit memory management. With only three times as much memory, it runs on average 17% slower than explicit memory management. However, with only twice as much memory, garbage collection degrades performance by nearly 70%. When physical memory is scarce, paging causes garbage collection to run an order of magnitude slower than explicit memory management.

Runtime Specialization With Optimistic Heap Analysis

Ajeet Shankar, UC Berkeley
Rastislav Bodik, UC Berkeley
Subramanya Sastry, UW Madison
James Smith, UW Madison

Runtime specialization is an optimization process that can provide significant speedups by exploiting a program's runtime state. Existing specializers employ a staged model that statically identifies specialization code regions and constant heap locations, and then optimizes them at runtime once the constants are known. This paper describes a dynamic program specializer that is transparent, and thus does not require any static components such as programmer annotations or pre-runtime analysis. The specializer uses (1) a novel store profile to identify constant heap locations that a staged system cannot, such as constants present in an interpreted program. This information is used in tandem with (2) a new algorithm for finding specialization starting points based on a notion of influence to create specializations. (3) An automatic invalidation transformation efficiently monitors assumptions about heap constants, and is able to deactivate specialized traces even if they are on the stack. An implementation of the specializer in Jikes RVM has low overhead in practice, selects specialization points that would be chosen manually, and produces speedups of 1.2x to 6.3x on a variety of benchmarks.

Tracing Traces **Thursday, 8:30-10:00** ***San Diego Room***

Session Chair: Cristina Videira Lopes, UC, Irvine

Adding Trace Matching with Free Variables to AspectJ

Chris Allan, Programming Tools Group, Oxford University
Pavel Avgustinov, Programming Tools Group, Oxford University
Aske Simon Christensen, BRICS, University of Aarhus
Laurie Hendren, Sable Research Group, McGill University
Sascha Kuzins, Programming Tools Group, Oxford University
Ondrej Lhoták, Sable Research Group, McGill University
Oege de Moor, Programming Tools Group, Oxford University
Damien Sereni, Programming Tools Group, Oxford University
Ganesh Sittampalam, Programming Tools Group, Oxford University
Julian Tibble, Programming Tools Group, Oxford University

An aspect observes the execution of a base program; when certain actions occur, the aspect runs some extra code of its own. In the AspectJ language, the observations that an aspect can make are confined to the current action: it is not possible to directly observe the history of a computation. Recently there have been several interesting proposals for new history-based language features, most notably by Douence et al, and also by Walker and Viggers. In this paper we present a new history-based language feature called tracematches, where the programmer can trigger the execution of extra code by specifying a regular pattern of events in a computation trace. We have fully designed and implemented tracematches as a seamless extension to AspectJ. A key innovation in our tracematch approach is the introduction of free variables in the matching patterns. This enhancement enables a whole new class of applications where events can be matched not only by the event kind, but also by the values associated with the free variables. We provide several examples of applications enabled by this feature. After introducing and motivating the idea of tracematches via examples, we present a detailed semantics of our language design, and we derive an implementation from that semantics. The implementation has been realised as an extension of the abc compiler for AspectJ.

Finding Application Errors Using PQL: A Program Query Language

Michael Martin, Stanford University
Monica Lam, Stanford University
Benjamin Livshits, Stanford University

A number of effective error detection tools have been built in recent years to check if a program conforms to certain design rules. An important class of design rules deals with sequences of events associated with a set of related objects. This paper presents a language called PQL (Program Query Language) that allows programmers to express such questions easily in an application-specific context. A query looks like a code excerpt corresponding to the shortest amount of code that would violate a design rule. Details of the target application's precise implementation are abstracted away. The programmer may also specify actions to perform when a match is found, such as recording relevant information or even correcting an erroneous execution on the fly.

We have developed both static and dynamic techniques to find solutions to PQL queries. Our static analyzer finds all potential matches conservatively using a context-sensitive, flow-insensitive, inclusion-based pointer alias analysis. Static results are also useful in reducing the number of instrumentation points for dynamic analysis. Our dynamic analyzer instruments the source program to catch all violations precisely as the program runs and optionally to perform user-specified actions.

We have implemented techniques described in this paper and used this combination of static and dynamic analysis to successfully find 206 breaches of security and important resource leaks in 6 large real-world open-source Java applications containing a total of nearly 60,000 classes.

Relational Queries Over Program Traces

Simon Goldsmith, UC Berkeley
Alex Aiken, Stanford University
Robert O'Callahan, Novell (formerly IBM T.J. Watson)

Instrumenting programs with code to monitor runtime behavior is a common technique for profiling and debugging. In practice, instrumentation is either inserted manually by programmers, or automatically by specialized tools that monitor particular properties. We propose Program Trace Query Language (PTQL), a language based on relational queries over program traces, in which programmers can write expressive, declarative queries about program behavior. We also describe our compiler, Partiqle. Given a PTQL query and a Java program, Partiqle instruments the program to evaluate the query on-line. We apply several PTQL queries to a set of benchmark programs, including the Apache Tomcat Web server. Our queries reveal significant performance bugs in the jack SpecJVM98 benchmark, in Tomcat, and in the IBM Java class library, as well as some correct though uncomfortably subtle code in the Xerces XML parser. We present performance measurements demonstrating that our prototype system has usable performance.

Concurrency || Concurrency **Thursday, 10:30-12:00** ***San Diego Room***

Session Chair: Dirk Riehle

Formalising Java RMI with Explicit Code Mobility

Alexander Ahern, Department of Computing, Imperial College London

Nobuko Yoshida, Department of Computing, Imperial College London

This paper presents a Java-like core language with primitives for object-oriented distribution and explicit code mobility. We apply our formulation to prove the correctness of several optimisations for distributed programs. Our language captures crucial but often hidden aspects of distributed object-oriented programming, including object serialisation, dynamic class downloading and remote method invocation. It is defined in terms of an operational semantics that concisely models the behaviour of distributed programs using machinery from calculi of mobile processes. Type safety is established using invariant properties for distributed runtime configurations. We argue that primitives for explicit code mobility offer a programmer fine-grained control of type-safe code distribution, which is crucial for improving the performance and safety of distributed object-oriented applications.

Lifting Sequential Graph Algorithms for Distributed-Memory Parallel Computation

Douglas Gregor, Indiana University

Andrew Lumsdaine, Indiana University

This paper describes the process used to extend the Boost Graph Library (BGL) for parallel operation with distributed memory. The BGL consists of a rich set of generic graph algorithms and supporting data structures, but it was not originally designed with parallelism in mind. In this paper, we revisit the abstractions comprising the BGL in the context of distributed-memory parallelism, lifting away the implicit requirements of sequential execution and a single shared address space. We illustrate our approach by describing the process as applied to one of the core algorithms in the BGL, breadth-first search. The result is a generic algorithm that is unchanged from the sequential algorithm, requiring only the introduction of external (distributed) data structures for parallel execution. More importantly, the generic implementation retains its interface and semantics, such that other distributed algorithms can be built upon it, just as algorithms are layered in the sequential case. By characterizing these extensions as well as the extension process, we develop general principles and patterns for using (and reusing) generic, object-oriented parallel software libraries. We demonstrate that the resulting algorithm implementations are both efficient and scalable with performance results for several algorithms.

Safe Futures for Java

Adam Welc, Purdue University

Antony Hosking, Purdue University

Suresh Jagannathan, Purdue University

A future is a simple and elegant abstraction that allows concurrency to be expressed often through a relatively small rewrite of a sequential program. In the absence of side-effects, futures serve as benign annotations that mark potentially concurrent regions of code. Unfortunately, when computation relies heavily on mutation as is the case in Java, its meaning is less clear, and much of its intended simplicity lost.

This paper explores the definition and implementation of safe futures for Java. One can think of safe futures as truly transparent annotations on method calls, which designate opportunities for concurrency. Serial programs can be made concurrent simply by replacing standard method calls with future invocations. Most significantly, even though some parts of the program are executed concurrently and may indeed operate on shared data, the semblance of serial execution is nonetheless preserved. Thus, program reasoning is simplified since data dependencies present in a sequential program are not violated in a version augmented with safe futures.

Besides presenting a programming model and API for safe futures, we formalize the safety conditions that must be satisfied to ensure equivalence between a sequential Java program and its future-annotated counterpart. A detailed implementation study is also provided. Our implementation exploits techniques such as object versioning and task revocation to guarantee necessary safety conditions. We also present an extensive experimental evaluation of our implementation to quantify overheads and limitations. Our experiments indicate that for programs with modest mutation rates on shared data, applications can use futures to profitably exploit parallelism, without sacrificing safety.

Exceptional Exceptions

Thursday, 13:30-15:00 ***San Diego Room***

Session Chair: Dave Thomas, Bedarra

Combining the Robustness of Checked Exceptions with the Flexibility of Unchecked Exceptions using Anchored Exception Declarations

Marko van Dooren, Katholieke Universiteit Leuven
Eric Steegmans, Katholieke Universiteit Leuven

Ever since their invention 30 years ago, checked exceptions have been a point of much discussion. On the one hand, they increase the robustness of software by preventing the manifestation of unanticipated checked exceptions at run-time. On the other hand, they decrease the adaptability of software because they must be propagated explicitly, and must often be handled even if they cannot be signalled.

We show that the problems with checked exceptions are caused by a lack of expressiveness of the exceptional return type of a method, which currently dictates a copy & paste style. We add the required expressiveness by introducing anchored exception declarations,which allow the exceptional behavior of a method to be declared relative to that of others. We present the formal semantics of anchored exception declarations, along with the necessary rules for ensuring compile-time safety, and give a proof of soundness.We show that anchored exception declarations do not violate the principle of information hiding when used properly, and provide a guideline for when to use them.

We have implemented anchored exception declarations in Cappuccino,which is an extension to the ClassicJava programming language.

Incrementalization Across Object Abstraction

Annie Liu, SUNY Stony Brook
Tom Rothamel, SUNY Stony Brook
Scott Stoller, SUNY Stony Brook
Yanni Liu, U of Manitoba
Michael Gorbovitski, SUNY Stony Brook

Object abstraction supports the separation of what operations are provided by systems and components from how the operations are implemented, and is essential in enabling the construction of complex systems from components. Unfortunately, clear and modular implementations have poor performance when expensive query operations are repeated, while efficient implementations that incrementally maintain these query results are much more difficult to develop and to understand, because the code blows up significantly and is no longer clear or modular.

This paper describes a powerful and systematic method that first allows the “what” of each component to be specified in a clear and modular fashion and implemented straightforwardly in an object-oriented language; then analyzes the queries and updates, across object abstraction, in the straightforward implementation; and finally derives the sophisticated and efficient “how” of each component by incrementally maintaining the results of repeated expensive queries with respect to updates to their parameters. Our implementation and experimental results for example applications in query optimization, role-based access control, etc. demonstrate the effectiveness and benefit of the method.

PolyD: A Flexible Dispatching Framework

Antonio Cunei, Purdue University
Jan Vitek, Purdue University

The standard dispatching mechanisms built into programming languages are sometimes inadequate to the needs of the programmer. In the case of Java, the need for more flexibility has led to the development of a number of tools, including visitors and multi-method extensions, that each add some particular functionality, but lack the generality necessary to support user-defined dispatching mechanisms. In this paper we advocate a more modular approach to dispatching, and we present a tool, PolyD, that allows the programmer to design custom dispatching strategies and to parametrize many aspects of the dispatching process. PolyD exhibits excellent performance and compares well against existing tools.

Student Research Competition

Chair: Dirk Siebert

After its remarkable success in the previous years, OOPSLA is hosting once more an ACM SIGPLAN Student Research Competition, sponsored by Microsoft.
The first round of evaluation will be held jointly with the OOPSLA PosterProgram Session. The ACM SIGPLAN Student Research Competition shares the Poster session's goal to facilitate students' interaction with researchers and to provide both sides with the opportunity to learn of ongoing, current research. Additionally, the Student Research Competition affords students with experience with both formal presentations and evaluations.

Presentation of Papers	**Monday, 17:30-19:30**	***Terrace Pavilion & Poolside***
Papers on Display	**Tuesday, 10:00-17:00**	***Terrace Pavillion***
Papers on Display	**Wednesday, 10:00-17:00**	***Terrace Pavillion***
Finalists' Presentation	**Wednesday, 10:30-12:00**	***Royal Palm 1 and 2***
Awards Presentation	**Thursday, 8:30-8:35**	***Golden West Room***
Papers on Display	**Thursday, 10:00-14:00**	***Terrace Pavillion***

Optimisation of Service Provision for Composite Web Services (CWS)
Ruth Lennon, *University College Dublin*

Grammar-Driven Generation of Domain-Specific Language Testing Tools
Hui Wu, *The University of Alabama at Birmingham*

Software Architecture Improvement through Test-Driven Development
David Janzen, *University of Kansas*

Formal Refactorings for Object Models
Rohit Gheyi, *Federal University of Pernambuco*
Tiago Massoni, *Federal University of Pernambuco*

A Software Product Line Architecture for Distributed Real-time and Embedded Systems: A Separation of Concerns Approach
Shih-Hsi Liu, *University of Alabama at Birmingham*

A Semi-Automated Approach for Analyzing, Separating, and Modeling of Concerns in Evolving Systems
Haitham Hamza, *University of Nebraska-Lincoln*

Metamodel-Driven Model Interpreter Evolution
Jing Zhang, *The University of Alabama at Birmingham*

Pattern Transformation for Two-Dimensional Separation of Concerns
Xiaoqing Wu, *Department of Computer and Information Sciences, The University of Alabama at Birmingham*

Supporting distributed software design meetings: What can we learn from co-located meetings?
Uri Dekel, *ISRI-SCS, Carnegie Mellon University*

Inferring Context-Free Grammars for Domain-Specific Languages
Faizan Javed, *University of Alabama at Birmingham*

Using Refactorings to Automatically Update Component-Based Applications
Danny Dig, *University of Illinois*

Autonomous Optmisation of Application Servers
John Bergin, *Department of Computer Science, University College Dublin, Belfield, Dublin*
Liam Murphy, *Department of Computer Science, University College Dublin, Belfield, Dublin*

Chair: Joe Bergin, Pace University

Take advantage of OOPSLA tutorials! If you think you're too inexperienced, or too experienced!, to benefit from OOPSLA tutorials, please think again! OOPSLA gathers the world's finest educators covering the full breadth of classic and cutting-edge topics. Our presenters not only have world-class expertise, they're also successful presenters who know how to share their hard-won knowledge with you. OOPSLA will provide 58 tutorials to choose from. It's a pity that the maximum that you can take is eight! Please take some time and carefully peruse the opportunities. The time you spend selecting your tutorials will certainly be repaid. There are, of course, many strong tutorials on classic OOPSLA topics like patterns, agile methods, Eclipse, programming languages, and aspect oriented programming. There are tutorials on functional and object-oriented programming, even scripting. There are design patterns and organizational patterns. You can find requirements, testing, concurrency, modelling, programming techniques, libraries, data base, theory, educational topics, web services, and many others. The special tutorials this year include two on Sunday by George Platts. His morning tutorial is in conjunction with Onward! and is primarily concerned with creativity. The second, Querdenking, will explore lateral thinking and how you might apply it to your work. Some of you know George, who is games-master at EuroPLoP and is an important contributor to the sense of community there. We also have two tutorials on Wiki. Look for the Intellectual Property tutorial on Wednesday, also.

17. Essential Object-Oriented Analysis and Design

Sunday, 8:30-17:00 ***Pacific Salon 5***

BEGINNER: This tutorial is targeted to people who are new to objects and object-oriented concepts. It is intended for people who have had some exposure to objects but need more knowledge to be able to put all the pieces together.

Jill Aden, EDS

Jill Aden, a Systems Architect, has worked with EDS since 1985 where she has worked with object-oriented technology since 1993. She has been the president of the Twin Cities Object Technology User Group (OTUG). Aden served as the Communications Chair at OOPSLA 2002 and was a tutorial presenter at ECOOP 2003 and OOPSLA 2002, 2003, and 2004. At EDS, Aden mentors, consults, and teaches object-oriented concepts internally and externally to clients. Jill is a Sun Certified Java Programmer.

Joseph Brennan, EDS

Joseph Brennan, a Consultant Architect working with EDS, has been working in the software industry since the mid 1980s. He has presented on object-oriented topics at user groups and conferences such as OOPSLA and ECOOP. He leads teams in all technical aspects of designing and deploying n-tier web based business applications globally. He works both at the strategic and hands-on levels as needed. As a part-time Field Instructor, he has instructed different levels of courses in the Java Platform. Joseph is a OMG UML Certified Professional, Sun Certified Java Instructor and Java Developer.

This tutorial is a short, quick-paced introduction to object-oriented analysis and design, based on practical project experience. It will provide you with the knowledge and skills to:

- Create use case documents and UML use case diagrams
- Understand object-oriented concepts, terminology and buzzwords
- Identify classes and create UML class diagrams
- Identify behaviors and create UML sequence diagrams, and recognize other UML diagrams
- Gain an understanding of what patterns are and review 5 of the "Gang of Four" design patterns

18. Software Factories

Sunday, 8:30-17:00 ***Pacific Salon 3***

ADVANCED: Attendees should be competent practitioners of current software development methods, practices and technologies. Familiarity with software product line practices, model driven development and component based development methods will be helpful.

Jack Greenfield, Microsoft Corporation

Jack Greenfield is an Architect for Enterprise Frameworks and Tools at Microsoft. He was previously Chief Architect, Practitioner Desktop Group, at Rational Software Corporation, and Founder and CTO of InLine Software Corporation. At NeXT Computer, he developed the Enterprise Objects Framework, now a part of Web Objects form Apple Computer. A well known speaker and writer, he is a co-author of the book "Software Factories: Assembling Applications with Patterns, Models, Frameworks and Tools", with Keith Short, Steve Cook and Stuart Kent. He has also contributed to UML, J2EE and related OMG and JSP specifications. He holds a B.S. in Physics from George Mason University.

Steve Cook, Microsoft Corporation

Steve Cook is a Software Architect in the Enterprise Frameworks and Tools group at Microsoft, which he joined at the beginning of 2003. Previously he was a Distinguished Engineer at IBM. He has worked in the IT industry for 30 years, as architect, programmer, author, consultant and teacher. He was one of the first people to introduce object-oriented programming into the UK, and has concentrated on languages, methods and tools for modeling since the early 1990s. He is a member of the Editorial Board of the Software and Systems Modeling Journal, a Fellow of the British Computer Society, and holds an Honorary Doctor of Science degree from De Montford University.

Assembling Applications with Patterns, Models, Frameworks and Tools

Increasingly complex and rapidly changing requirements and technologies are making development increasingly difficult. Promising advances have been made, however, in component based and model driven development, software architecture, aspect oriented programming, and requirements, process and software product line engineering. This tutorial presents Software Factories, a paradigm for automating software development that integrates these advances to increase agility, productivity, and predictability across the software life cycle. We will show a worked example of a software factory and perform small group exercises that help participants explore this approach. Participants will learn about the software factory schema, a graph of viewpoints used to separate concerns, relating work done at one level of abstraction, in one part of a system, or in one phase of the life cycle, to work done at other levels, or in other parts and phases, and about how the schema can be used to deliver guidance and to support its enactment through model transformation, constraint checking and other techniques. We will also describe the software factory life cycle and show how software factories can be specialized and composed. Finally, we will discuss software supply chains and show how Software Factories compose across organizational boundaries.

GP1. How has the arts, sports or life stimulated, inspired and informed your work in computer science?

Sunday, 8:30-12:00 ***Terrace Pavilion and Poolside***

BEGINNER: Interest and an open mind is all that is needed.

George Platts, Gamesmaster

George is a multi medium artist (performance, video and sound installations), Art Psychotherapist for the UK National Health Service (Psychiatry), narrowboat owner and cyclist "Living as a Dutchman in England". From 1993, working as an Artist-in-Residence / Tangential Thinking Co-ordinator / Querdenker Koordinator (Germany) at conferences of the Pattern Languages of Programming community in the US and Europe.

Computer science, especially software design, interacts with the arts and other life experiences; mathematics and music are closely linked. Since 1993 I have been exploring artists' patterns with the patterns community at PLoP conferences in USA, Germany, Scandinavia and Japan. This interactive tutorial at OOPSLA, will offer an opportunity for delegates to share such inspirational moments. On my part, I will be exhibiting a wealth of cross-arts' material that has excited PLoP computer scientists since 1993; artists' movies and film from France, Canada and USA; unusual music from Australia, Egypt, The Netherlands and USA.

This tutorial is in conjunction with the Onward! track of OOPSLA 2005.

1. Patterns in Functional Programming

Sunday, 8:30-12:00 ***Pacific Salon 6***

INTERMEDIATE: Fluency in OO programming. Familiarity with design patterns. No experience in functional programming is required, but confidence with recursion is essential.

Jeremy Gibbons, University of Oxford

Jeremy Gibbons is University Lecturer in Software Engineering and Continuing Education at the University of Oxford. He teaches on the professional part-time Masters programme (www.softeng.ox.ac.uk), and acts as Deputy Director with particular responsibility for the Object Technology sub-programme. He has eighteen years' experience in research and teaching in programming languages, especially in functional programming.

The purpose of this tutorial is to draw together ideas from the Design Patterns community (the Gang of Four: Gamma, Helm, Johnson, Vlissides) and the Functional Programming world (eg Bird, Meertens, Hughes). In particular, the thesis is that whereas design patterns must be expressed extra-linguistically (as prose, diagrams, examples) in object-oriented languages, they may be captured directly as abstractions using higher-order operators in functional programming languages. Therefore, they may be reasoned about, type-checked, applied and reused, just as any other abstractions may be.

We argue this case by developing the idea of higher-order operators, specifically for capturing patterns of computation in programs. We then build on this to show how the intentions behind a number of the Gang of Four patterns - such as Composite, Visitor, Iterator, and Builder - have higher-order operators as their analogues in functional languages. Specifically, the structure of these patterns is determined essentially by the structure of the data involved, and they can be captured as generic programs parametrized by that datatype.

The aim is to give greater insight into and understanding of already-familiar patterns.

2. Tuning Your Methodology to You

Sunday, 8:30-12:00 ***Pacific Salon 2***

ADVANCED: Experienced people trying to choose or tune a methodology, who have used and thought about at least one methodology.

Alistair Cockburn, Humans and Technology

Alistair Cockburn, founder of Humans and Technology, was special advisor to the Central Bank of Norway for object technology and software project management, the OO methodology designer for the IBM Consulting Group in the 1990s. In 2001 he co-authored the Agile Software Development Manifesto, and in 2005 co-authored the related project management Declaration of Inter-Dependence. Several of his books "Surviving OO Projects", Writing Effective Use Cases, Agile Software Development, and Crystal Clear: A Human-Powered Methodology for Small Teams. Articles, talks and supporting materials can be found at http://alistair.cockburn.us.

A methodology is a social construction of an organization. The first part of the tutorial introduces language and constructs needed to evaluate, compare and construct methodologies. These include precision, accuracy, tolerance, relevance, and scale, along with the nine basic elements of a methodology. Several examples of effective, lightweight and real methodologies are given, along with commentary on the social setting for each. The tutorial examines the conditions suited to shifting from a lighter to a heavier methodology and the penalty for doing so. The tutorial ends with the presentation of a small family of lightweight and practical methodologies, optimized for productivity, making maximum use of human, face-to-face communication. Considerations about success and failure in affecting culture are visited again at the end. Learn to identify and diagnose the parts of your organization's methodology, and learn ways to make it more effective.

3. Enterprise Aspect-Oriented Programming with AspectJ

Sunday, 8:30-12:00 ***Pacific Salon 7***

BEGINNER: Attendees should have enterprise Java development experience.

Ronald Bodkin, New Aspects of Software

Ron Bodkin is the founder of New Aspects of Software, which provides consulting and training on application development with an emphasis on aspect-oriented programming and performance management. Ron is also a member of AspectMentor, a consortium of AOP experts. Ron speaks and gives tutorials frequently for customers and at conferences. Previously, Ron led the first implementation projects and training for customers for the AspectJ group at Xerox PARC. Prior to that, Ron was a founder and the CTO of C-bridge, a consultancy that delivered enterprise applications using Java frameworks.

Aspect-Oriented Programming (AOP) has become more visible than ever for enterprise development with support from IBM, BEA, Oracle, JBoss, and Spring. AspectJ is a seamless AOP extension to Java, and is used to teach the concepts. It allows consistent and flexible implementation of crosscutting concerns such as metering, error handling, testing, and yes logging. This is a major improvement on scattered and tangled traditional implementations.

This tutorial teaches developers how to develop with aspects and lets them see in depth examples of where it can be used in the enterprise. Attendees see how to use AOP to solve a range of problems in enterprise Java application development, such as tracing, testing, error handling, and business aspects.

Attendees see demonstrations of how tools support (in Eclipse) allows developers to see how aspects interact with a system, and how to integrate aspects into real-world J2EE environments. Attendees learn the core concepts of AOP, see practical applications, gain insight into effective ways to design, develop and adopt aspects. The tools used in the tutorial are all freely available as open source software, so participants will be able to use the techniques shown in their own projects.

4. Code Smells

Sunday, 8:30-12:00 ***Sunset***

INTERMEDIATE: Attendees should understand and regularly use OO concepts: encapsulation, polymorphism, and loose coupling.

Ken Scott-Hlebek, Industrial Logic, Inc.

Ken Scott-Hlebek is an Industrial XP Coach. Prior to joining Industrial Logic, Ken assisted teams in their transition to agility by teaching test-driven development, refactoring, and evolutionary design. Ken brings experience as a software craftsman and mentor from a variety of environments, from small in-house teams to large outsourced and offshore projects. Convinced that human factors are the key to successful software teams, Ken is passionate about promoting the warm and fuzzy side of the craft.

A "code smell" is a design problem. Code smells are found in methods, classes, hierarchies, packages (namespaces, modules) or entire systems. To be good at refactoring — which means "improving the design of existing code" — you must learn how to spot code that needs improvement. This tutorial will help you do that. We'll explore a comprehensive set of code smells from the books, Refactoring and Refactoring to Patterns. We'll look at real world examples of code smells, and we'll discuss which refactorings are most often used to deodorize code.

5. Adding Software Testing to Programming Assignments

Sunday, 8:30-12:00 ***Pacific Salon 4***

BEGINNER: Educators interested in testing or XP. No specific testing knowledge required. Basic Java suggested but not required.

Stephen Edwards, Virginia Tech, Dept. of Computer Science

Stephen Edwards is an associate professor in the Department of Computer Science at Virginia Tech, with research interests in software engineering, reuse, automated testing, and CS education. He is a vocal proponent of teaching software testing across the CS curriculum. He recently redesigned the CS1 course at Virginia Tech, where students begin writing simple tests for their programs during the first week, and must submit tests along with every lab and programming assignment. These techniques have been applied in freshman, sophomore, and upper division undergraduate courses, as well as in graduate courses and at other universities.

Software testing is a critical skill for practitioners, but it is rarely given appropriate coverage in undergraduate curricula. If we want to change the culture of how students program, they should practice basic testing skills on every assignment all the time. This tutorial provides a practical introduction to incorporating software testing activities as a regular part of programming assignments. While this can be done for many languages, the focus here is on leveraging XUnit-style tools and test-first coding techniques to infuse software testing throughout courses that teach object-oriented programming concepts. Empirical evidence suggests that, when assignments are transformed this way, students produce higher-quality results (averaging 28% fewer bugs/KSLOC) and are more likely to turn assignments in on time. Further, test-first strategies preempt "big bang" integration disasters, and students report greater confidence in their own work.

This tutorial presents five different models for how one can incorporate testing into assignments, provides live programming demonstrations of the techniques, and discusses the corresponding advantages and disadvantages of each. Approaches to assessment (using testing to assess student code, assessing tests that students write, and automated grading) are all discussed. Advice for writing "testable" assignments is given. Hands-on examples for participants are also provided.

6. Working Effectively with Legacy Code

Sunday, 8:30-12:00 ***Sunrise***

INTERMEDIATE: Each participant should be an experienced C, C++, Java, or C# programmer, with first-hand experience attempting to modify legacy code written in one of those languages. Particpants should also have reading knowledge of Java and C++.

Michael Feathers, Object Mentor

Michael works with Object Mentor as mentor and consultant. He is the author of 'Working Effectively with Legacy Code.' He is also the original author of CppUnit, a C++ port of the JUnit testing framework, and CppFit, a C++ port of the FIT integrated-testing framework. A member of ACM and IEEE, he has chaired CodeFest at several OOPSLA conferences.

Test Driven Development and Refactoring are powerful tools in the software development arsenal. With them you can add new code to systems and make existing code more maintainable. However, refactoring code without having tests in place can be hazardous. This tutorial presents a collection of dependency breaking and test writing techniques that can be used to get existing code safely under test for refactoring and enhancement. These techniques can be used in conjunction with Test Driven Development to breathe new life into large existing code bases.

7. Generative Software Development

Sunday, 8:30-12:00 ***Esquire***

INTERMEDIATE: This tutorial is aimed at researchers and practitioners interested in modeling and implementation technologies to achieve reusability and automation.

Krzysztof Czarnecki, University of Waterloo

Krzysztof Czarnecki is an Assistant Professor at the University of Waterloo, Canada. Before coming to Waterloo, he spent 8 years at DaimlerChrysler Research working on the practical applications of generative programming. He is co-author of the book "Generative Programming" (Addison-Wesley, 2000), which is regarded as founding work of the area and is used as a graduate text at universities around the world. He was General Chair of the 2003 International Conference on Generative Programming and Component Engineering (GPCE) and keynote speaker at UML 2004. His current work focuses on realizing the synergies between generative and model-driven software development.

Product-line engineering seeks to exploit the commonalities among systems from a given problem domain while managing the variabilities among them in a systematic way. In product-line engineering, new system variants can be rapidly created based on a set of reusable assets (such as a common architecture, components, models, etc.). Generative software development aims at modeling and implementing product lines in such a way that a given system can be automatically generated from a specification written in one or more textual or graphical domain-specific languages (DSLs).

In this tutorial, participants will learn how to perform domain analysis (i.e., capturing the commonalities and variabilities within a system family in a software schema using feature modeling), domain design (i.e., developing a common architecture for a system family), and implementing software generators using multiple technologies, such as template-based code generation and model transformations. Available tools for feature modeling and implementating DSLs as well as related approaches such as Software Factories and Model-Driven Architecture will be surveyed and compared. The presented concepts and methods will be demonstrated using a sample case study of an e-commerce platform.

8. Adaptive Object-Model Architecture

Sunday, 8:30-12:00 ***Pacific Salon 1***

ADVANCED: A good knowledge of object concepts is required. It would be useful if participants have a basic understanding of frameworks, though it is not necessary. A general understanding of the GOF patterns is required. An understanding of Analysis Patterns and Reflective Architectures can be helpful. Specifically we will be covering Composite, TypeObject, Properties, Strategy, Interpreter and the Builder Design Patterns along with the Party, Accountability and Observation Analysis Patterns.

Joseph Yoder, The Refactory, Inc.

Joseph W. Yoder from The Refactory, Inc., has worked on the architecture, design and implementation of software projects dating back to 1985. These projects range from stand-alone to client-server applications, multi-tiered, databases, object-oriented, frameworks, human-computer interaction, collaborative environments, web-based, and domain-specific visual-languages. Joe is the author of over two-dozen published patterns and has been working with patterns for a long time, writing his first pattern paper in 1995. Recently Joseph's focus has been on how to build dynamic and adaptable systems and he has been providing analysis, design, and mentoring along with writing papers to reflect these experiences.

How to Build Systems That Can Dynamically Adapt to Changing Requirements

Architectures that can dynamically adapt to changing requirement are sometimes called "reflective" or "meta" architectures. We call a particular kind of reflective architecture an "Adaptive Object-Model (AOM)" architecture. An Adaptive Object-Model is a system that represents classes, attributes, relationships, and behavior as metadata. It is a model based on instances rather than classes. Users change the metadata (object model) to reflect changes to the domain model. These changes modify the system's behavior. In other word, it stores its Object-Model in XML files or in a database and interprets it. Consequently, the object model is adaptive; when the descriptive information for the object model is changed, the system immediately reflects those changes. We have noticed that the architects of a system with Adaptive Object-Models often claim this is the best system they have ever created, and they brag about its flexibility, power, and eloquence. At the same time, many developers find them confusing and hard to work with. This is due in part because the developers do not understand the architecture. This tutorial will give a description of the Adaptive Object-Model architectural style and will make it easier for developers to understand and build systems that need to adapt to changing requirements.

GP2. Querdenking: Can creativity be taught?

Sunday, 13:30-17:00 ***Terrace Pavilion and Poolside***

BEGINNER: Interest and an open mind is all that is needed.

George Platts, Gamesmaster

George is a multi medium artist (performance, video and sound installations), Art Psychotherapist for the UK National Health Service (Psychiatry), narrowboat owner and cyclist "Living as a Dutchman in England". From 1993, working as an Artist-in-Residence / Tangential Thinking Co-ordinator / Querdenker Koordinator (Germany) at conferences of the Pattern Languages of Programming community in the US and Europe.

Edward de Bono, the originator of the term Lateral Thinking, believes that creativity can be taught (see "Lateral Thinking : A Textbook of Creativity" 1990). Employed as a Tangential Thinking Co-ordinator at PLoP conferences in Illinois since 1995 and as a Querdenker Koordinator at EuroPLOP since 1996 (and also in Scandinavia - VikingPLOP- and Japan -MensorePLoP) I have facilitated a variety of lateral activities that have relevance to software designers / computer scientists. These activities have been informed not only by Lateral Thinking concepts but also by the New Games Foundation (originated by Stewart Brand and George Leonard in San Francisco in 1973) and by the work of a variety of artists, composers and movie makers.

9. Steps to an Agile Frame of Mind - the First Three Hours

Sunday, 13:30-17:00 ***Pacific Salon 2***

BEGINNER: Anyone curious about agile development, for software or even not. The workshop is non-technical in the sense that it is non-programming; it is technical in that it contains material that is core to the modern life of professional developers, project managers, and business managers. Because the issues cover all aspects and stakeholders in the development process, this workshop is recommended for business executives, project managers, business analysts and developers alike.

Alistair Cockburn, Humans and Technology

Alistair Cockburn, founder of Humans and Technology, was special advisor to the Central Bank of Norway for object technology and software project management, the OO methodology designer for the IBM Consulting Group in the 1990s. In 2001 he co-authored the Agile Software Development Manifesto, and in 2005 co-authored the related project management Declaration of Inter-Dependence. Several of his books "Surviving OO Projects", Writing Effective Use Cases, Agile Software Development, and Crystal Clear: A Human-Powered Methodology for Small Teams. Articles, talks and supporting materials can be found at http://alistair.cockburn.us.

Agile is an *attitude* giving priority to communication efficiency, feedback and reflective improvement. This attitude produces process efficiency and maneuverability with respect to changing requirements, technology and team. Having the agile attitude does not yet make for a successful project.

This workshop, based on the award-winning book, Agile Software Development, is a three-hour mixture of lecture, exercises, and discussion. The purpose of these three hours is to give attendees a sense for how it feels to be doing agile versus non-agile development and some words around key issues. Needless to say, in three hours the attendee will only see the first steps of the journey, but should come away with some of the crucial sensations and central ideas.

10. The Six Million Dollar Customer

Sunday, 13:30-17:00 ***Pacific Salon 6***

INTERMEDIATE: This tutorial is aimed at anyone who wants to improve the effectiveness of the "Customer" within XP. Participants should have a background in XP (or another Agile methodology).

Angela Martin, Victoria University of Wellington

Angela Martin is a PhD Candidate at Victoria University of Wellington, New Zealand, supervised by James Noble and Robert Biddle. Her research utilises in-depth case studies of the XP Customer Role, on a wide range of projects world-wide, complementing her 10 years of professional experience in software development.

Robert Biddle, Human-Oriented Technology Laboratory, Carleton University

Robert Biddle is Professor of Human-Oriented Technology at Carleton University in Ottawa, Canada. His research and teaching is in software engineering and human-computer interaction; he earlier worked as a software developer and as a technical consultant. He is widely recognised as an excellent teacher, presenter, and educator.

James Noble, Victoria University of Wellington

James Noble is Professor of Software Engineering at Victoria University of Wellington, New Zealand. He has extensive experience lecturing, teaching, and mentor-ing software design, software visualisation, user interface design, and design patterns, and many other topics. He has presented many tutorials at conferences including OOPSLA, JAOO, TOOLS, OzCHI, and VL/HCC.

This tutorial will introduce you to practices that will increase the effectiveness of "the customer" on your XP project. Customers have one of the most complex and difficult roles on a project, yet XP includes very few practices that support the customer in their role—other than prescribing how they interact with the developers. Over the last three years, we have investigated many projects around the world to identify how customers succeed in this complex and difficult task—discovering not what people think should have happened, but what really happened and what actually worked! This tutorial will present a consistent set of practices, skills, and characteristics that are crucial to ensuring your customer role, and thus your XP project, will be successful.

11. Teaching Java: An Eventful Approach

Sunday, 13:30-17:00 ***Pacific Salon 4***

INTERMEDIATE: Knowledge of basic Java programming. Experience teaching an introductory programming course is helpful, but not necessary.

Kim Bruce, Pomona College

Kim Bruce is Professor of Computer Science at Pomona College. He has taught at Princeton University and Williams College, and has been a visiting professor or scientist at M.I.T., Stanford, Ecole Normale Superieure, University of Pisa, the Newton Institute at Cambridge University, and UC Santa Cruz. He has presented papers at SIGCSE, ECOOP, OOPSLA, OOPSLA Educators' Symposium, and POPL, and is the author of Foundations of Object-Oriented Languages: Types and Semantics, published by MIT Press, and Java: An eventful approach, published by Prentice Hall. He received the SIGCSE 2005 award for outstanding contributions to Computer Science Education.

As many have learned, teaching a CS1 course in Java requires an entirely new approach. Because it is an object-oriented language, it is difficult to teach in the style used for imperative languages like Pascal, C, and C++. Many have proposed teaching Java in an objects-first way, but others have complained that too many concepts must be introduced before students can understand the construction of classes and objects. The objective of this workshop is to introduce an approach to teaching Java that we have developed that overcomes these problems.

In this tutorial we describe an objects-first approach to teaching Java that introduces event-driven programming in the very first programming examples, introduces concurrent threads early, and uses graphics and animation extensively.

- We show how these seemingly advanced topics can be presented so that they are easy for introductory course students to grasp.
- We also show how our approach exposes students to object-oriented programming techniques more thoroughly than is possible in more traditional approaches.
- Our approach is supported by materials developed with NSF funding including extensive course notes, laboratory exercises, the objectdraw library, and a textbook published by Prentice-Hall in the summer of 2005.

12. Use Cases Are Early Aspects

Sunday, 13:30-17:00 ***Sunrise***

INTERMEDIATE: Attendees should have some basic knowledge of aspects, use cases and UML

Ivar Jacobson, Ivar Jacobson Consulting

Ivar Jacobson, Ph.D., is "the father" of many technologies including components and component architecture, use cases, modern business engineering, and the Rational Unified Process. He was one of the three amigos who originally developed the Unified Modeling Language. He is the principal author of five best-selling books on these methods and technologies, in addition to being the coauthor of the two leading books on the Unified Modeling Language. Ivar is founder of Ivar Jacobson International, which has offices around the world.

Pan-Wei Ng, Ivar Jacobson Consulting

Pan-Wei Ng, Ph.D., plays multiple roles in Ivar Jacobson Consulting (IJC). As a member of the IJC Technology Office, Pan-Wei defines best practices in architecture, use cases, iterative development, business modeling, active software, and aspects. He develops materials for some of these best practices and conducts research in other best practice areas. Pan-Wei also actively works on customer accounts to enable companies and project teams to adopt these best practices quickly and safely. Pan-Wei works alongside practitioners and ensures that the best practices developed are both relevant and practical.

Everyone talks about aspects—but most people think about them as a programming technique to deal with low level crosscutting concerns. We know that aspects can be applied to large scale projects, but how do you use them in real software development from requirements to code? With use cases of course! A strong parallel exist between use cases and aspects in that just as aspects cut across objects, so does use cases. Furthermore, use cases are "early aspects". They provide the means to model stakeholder concerns, even crosscutting ones. Thanks to aspect-orientation techniques, you can keep these use cases or early aspects separate from each other all the way to code. This works for features of many kinds, functional or non-functional or infrastructural or even platform specifics. In doing so, we take a new grasp on Model Driven Architecture. Each "aspect" of the design model (i.e. the platform specific model) is very much like an overlay as used for an overhead projector. You stack them up to produce the whole system. We added new constructs needed to UML to model these overlays. In this way, we formulate a simpler and more aspect-oriented approach to architecture.

13. The Common Lisp Object System: Generic Functions and Metaobject Protocol

Sunday, 13:30-17:00 ***Pacific Salon 7***

ADVANCED: A good understanding of class-based OOP. Experience with may be helpful, but the tutorial is specifically targeted at non-Lispers.

Pascal Costanza, Vrije Universiteit Brussel

Pascal Costanza has a Ph. D. degree from the University of Bonn, Germany. His past involvements include specification and implementation of the languages Gilgul and Lava, and the design and application of the JMangler framework for load-time transformation of Java class files. He has also implemented aspect-oriented extensions for CLOS that heavily rely on its MOP, and currently explores possibilities for making object-oriented programs better adaptable to the context of their use. He is furthermore the initiator and lead of Closer, an open source project that provides a compatibility layer for the CLOS MOP across multiple Common Lisp implementations.

The Common Lisp Object System (CLOS) is unique in two ways. 1) In most OOP languages, methods belong to classes and are invoked by sending messages. In CLOS, methods belong to generic functions, and they select and execute the correct method according to the types of the arguments they receive. 2) The CLOS Metaobject Protocol (MOP) specifies how its essential building blocks are to be implemented in CLOS itself. This allows extending its object model with metaclasses that change important aspects of CLOS for a well-defined scope.

This tutorial introduces these two notions. I will develop—live during the tutorial—the code for an interpreter for generic functions that performs selection and execution of methods. I will then discuss how that code can be extended to introduce, for example, multimethods and AOP-like advices, and sketch how generic functions are implemented efficiently in the "real" world. In the second part, I will illustrate the extensibility of the CLOS MOP by implementing—live—the (hashtable-based) Python object model as a metaclass. Other practical extensions based on the CLOS MOP are also sketched, like object-relational mappings, interfaces to foreign-language objects, and domain-specific annotations in classes.

14. A Tour of Responsiblity-Driven Design

Sunday, 13:30-17:00 ***Pacific Salon 1***

INTERMEDIATE: Participants should be familiar with object concepts and be looking for practical techniques, guidelines and a design process that emphasizes modeling the behavioral aspects of a software system.

Rebecca Wirfs-Brock, Wirfs-Brock Associates

Rebecca Wirfs-Brock is a well-known and respected object practitioner and a lifetime attendee of OOPSLA. She invented the way of thinking about objects known as Responsibility-Driven Design and is the lead author of the classic Designing Object-Oriented Software (1990), and Object Design: Roles, Responsibilities and Collaborations (2003). Rebecca has been involved with object technology since the early days. Through her writing, teaching and speaking she has popularized the use of informal techniques and thinking tools for designers and analysts. Among her widely-used inventions are object role stereotypes and the conversational form of use cases. From development on the Tektronix implementation of Smalltalk in the early 1980's, through years of development and training experience, she is recognized as an innovative and influential practitioner of object-oriented design.

Responsibility-Driven Design is a way to design that emphasizes behavioral modeling using objects, responsibilities and collaborations. In a responsibility-based model, objects play specific roles and occupy well-known positions in the application architecture. Each object is accountable for a specific portion of the work. They collaborate in clearly defined ways, contracting with each other to fulfill the larger goals of the application. By creating a "community of objects", assigning specific responsibilities to each, you build a collaborative model of our application.

This tutorial, which includes material from the book Object Design: Roles, Responsibilities and Collaborations, will be an example-based tour of Responsibility-Driven Design. It presents our latest innovations and practical techniques. Topics include: an updated view of CRC card models, finding and evaluating the qualities of candidate design objects, designer's stories, mapping roles to classes and interfaces, strategies for assigning object responsibilities, deciding on the control style of an application, collaboration trust regions, and collaboration contracts. Attendees will have an opportunity to practice techniques with several short design exercises.

15. Storytest-Driven Development

Sunday, 13:30-17:00 ***Sunset***

INTERMEDIATE: Attendees should have a working knowledge of Java and a basic understanding of test driven development.

Max Baumann, Industrial Logic Inc.

Max Baumann has been programmer, coach, and trainer on small and large scale Traditional, Agile, and XP projects over the past ten years. Max is a capable developer who brings theory to practice by working side-by-side with teams to improve their collaboration, refactoring, test-driven development, and storytest-driven development skills. He regularly shares his expertise with organizations and conferences as a trainer and speaker. He is committed to refining his craft, and is passionate about making software development enjoyable for everyone from customers and programmers to the end users. In his spare time he can be found enjoying the outdoors, cooking and playing the mandolin.

Ken Scott-Hlebek, Industrial Logic, Inc.

Ken Scott-Hlebek is an Industrial XP Coach. Prior to joining Industrial Logic, Ken assisted teams in their transition to agility by teaching test-driven development, refactoring, and evolutionary design. Ken brings experience as a software craftsman and mentor from a variety of environments, from small in-house teams to large outsourced and offshore projects. Convinced that human factors are the key to successful software teams, Ken is passionate about promoting the warm and fuzzy side of the craft.

Falling behind on writing your acceptance tests? This is a common problem. Either you don't have the right people to produce acceptance tests and/or you just never seem to have enough time to implement them. The result is a backlog of work and not enough coverage for the features in your system. The solution to this problem is called Storytest-Driven Development (SDD).This practice advocates you define storytests (automated acceptance criteria) for stories prior to implementing them. Writing the storytest aids both developers and customers in aligning their efforts. Storytests are textual descriptions of a test that are easy to read, modify and execute. Subject matter experts, analysts, testers and programmers produce tests collaboratively. Storytests may be automated using a variety of technologies, such as FIT, Ward Cunngingham's excellent Framework for Integrated Test. A key benefit of this practice is the design simplicity that flows out of starting all work with a failing storytest. This tutorial will introduce you to the practice of SDD. We'll explore example code and discuss real-world applications of this practice. We'll also discuss strategies for writing Storytests on 'hard to test applications'.

16. Hands On AspectJ Development for Enterprise

Sunday, 13:30-17:00 ***Esquire***

INTERMEDIATE: Exposure to AOP concepts (e.g., by attending the introduction to enterprise AOP tutorial), Java knowledge, a laptop with Eclipse and AJDT AspectJ plug-in.

Ron Bodkin, New Aspects of Software

Ron Bodkin is the founder of New Aspects of Software, which provides consulting and training on application development with an emphasis on aspect-oriented programming and performance management. Ron is also a member of AspectMentor, a consortium of AOP experts. Ron speaks and gives tutorials frequently for customers and at conferences. Previously, Ron led the first implementation projects and training for customers for the AspectJ group at Xerox PARC. Prior to that, Ron was a founder and the CTO of C-bridge, a consultancy that delivered enterprise applications using Java frameworks.

Aspect-Oriented Programming (AOP) has become more visible than ever for enterprise development with support from IBM, BEA, Oracle, JBoss, and Spring. AspectJ is a seamless AOP extension to Java, and is used to teach the concepts. It allows consistent and flexible implementation of crosscutting concerns such as metering, error handling, testing, and yes logging. This is a major improvement on scattered and tangled traditional implementations.

This hands on tutorial lets developers gain experience developing with aspects by applying it to practical examples for enterprise use. Attendees apply AOP to solve a range of problems in enterprise Java application development, such as tracing, testing, error handling, and business aspects. Exercises are done using AspectJ in Eclipse.

Attendees experience developing with aspects and see how tools support lets one see how aspects interact with a system.

Attendees will learn both the potential and the pitfalls of applying AOP, and understand the basic mechanisms. The tools used in the tutorial are all freely available as open source software, so participants will be able to use the techniques shown in their own projects. At the end of the class, participants will have had experience developing with aspects.

35. Mastering UML with Stable Software Patterns

Monday, 8:30-17:00 ***Esquire***

INTERMEDIATE: Familiarity with basic notions of software engineering, UML notation and models, and design patterns.

Mohamed Fayad, San Jose State University

Dr. Fayad is a Full Professor of Computer Engineering at San Jose State University. He has 15+ years of industrial experience. He was a guest editor on nine theme issues, and has given tutorials and seminars on OO Technologies and Experiences at many conferences in USA and several countries. Dr. Fayad received an MS and a Ph.D. in computer science, from the University of Minnesota at Minneapolis. He is the lead author of several Wiley books: Transition to OO Software Development, August 1998, Building Application Frameworks, Implementing Application Frameworks, Domain-Specific Application Frameworks.

Haitham Hamza, University of Nebraska-Lincoln

Mr. Hamza received an MS in computer science, from the University of Nebraska-Lincoln, August 2002 and an MS in Electronics & Communication Engineering from Cairo University, December 2000. His research topic was A Foundation for Building Stable Analysis Patterns. He is a co-author of a new book on "Stable Software Patterns" with Dr. Fayad and Dr. Cline, Wiley 2005. He works for ActiveFrameworks as a lead researcher in stable software patterns. Mr. Hamza was the co-chair of several workshops organized in conjunction with IEEE IRI 2003, IEEE/ACM UML 2003, AICCSA 2005, ECOOP 2005, and IEEE IRI 2005.

The Unified Modeling Language (UML) has become the de facto modeling language for developing software systems and applications. As such, UML becomes an integral part of virtually every course in software development in both industry and academia. Personal experience with UML in the classroom and in professional organizations suggests that a basic understanding of UML notations and models is straightforward. However, the application of the correct model within context is not an easy task in practice. This is due to the fact that most existing resources tend to describe UML artifacts without providing the sufficient knowledge needed to learn when and how these artifacts should be best used. In this tutorial, we introduce a collection of pattern languages to capture, document, and communicate the core knowledge of both the syntax and the semantics of the different modeling artifacts in UML. This collection of pattern languages will serve as a base for abstraction, education, training, and research. Each pattern language provides a complete view of the necessarily knowledge to understand and effectively apply a specific modeling artifact. The use of the concept of pattern languages makes documenting and communicating the core knowledge easy to understand and retrieve when needed.

36. Pattern-Oriented Software Architecture

Monday, 8:30-17:00 ***Pacific Salon 1***

ADVANCED: The tutorial is intended for programmers familiar with OO development techniques and language features and systems programming and networking concepts.

Douglas Schmidt, Vanderbilt University

Dr. Douglas C. Schmidt is a Full Professor in the Electrical Engineering and Computer Science Department at Vanderbilt University. For over a decade, his research has focused on patterns, optimization techniques, and empirical analyses of object-oriented frameworks that facilitate the development of distributed real-time and embedded (DRE) middleware on parallel platforms running over high-speed networks and embedded system interconnects. In addition to his academic research, Dr. Schmidt has over 15 years of experience leading the development of ACE and TAO, which are widely used DRE middleware frameworks that contain a rich set of components that implement patterns for DRE systems.

Patterns for Concurrent and Distributed Systems

This tutorial describes how to apply patterns, frameworks, and middleware to alleviate the complexity of developing concurrrent and distributed applications. These patterns and frameworks have been used successfully by the speaker on production applications at hundreds of commercial companies and projects.

The tutorial illustrates by example how to significantly simplify and enhance the development of distributed systems that effectively utilizes concurrency and distribution via the use of:

- Object-oriented design techniques, such as patterns, layered modularity, and data/control abstraction
- Object-oriented language features, such as abstract classes, inheritance, dynamic binding, and parameterized types
- Middleware, such as object-oriented frameworks for distributed object computing and component middleware
- Advanced operating system mechanisms, such as event demultiplexing, multi-threading, multi-processing, and explicit dynamic linking

The tutorial examines patterns and framework solutions abstracted from production systems in domains ranging from telecommunication systems, medical systems, real-time avionics and aerospace systems, e-commerce, and automated stock trading to illustrate key technical design and implementation issues. The material presented in this tutorial is based on the book "Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects", Wiley & Sons, 2000, which is the second volume in the highly acclaimed POSA series.

19. Java Reflection

Monday, 8:30-12:00 ***Pacific Salon 4***

INTERMEDIATE: An attendee must be a competent Java programmer.

Ira Forman, IBM

Dr. Ira R. Forman works for IBM in Austin. As a member of IBM's Object Technology Products Group, which produced the SOMobjects Toolkit, he worked on the SOM Metaclass Framework. He started working in the area of object-oriented programming in 1984, when he worked at ITT Programming Technology Center. Forman received his Ph.D. in Computer Science from the University of Maryland, where he studied under Harlan Mills. Forman's specialties are object-oriented programming, distributed systems, and object composition. He is the coauthor of two books: "Interacting Processes: A Multiparty Approach to Coordinated Distributed Programming" and "Putting Metaclasses to Work: A New Dimension in Object-Oriented Programming."

Nate Forman, Ticom Geomatics

Nate Forman works for Ticom Geomatics where he designs and programs application frameworks for their products. His specialties are patterns and object-oriented programming. Forman holds a MSE in Software Engineering from the University of Texas at Austin and a BS in Computer Science from the College of Engineering at Cornell University.

The use of reflection is an important technique for improving program flexibility and programmer productivity. Reflection facilitates development of programs that are easily adapted to requirement changes. This property of reflection implies better code reusability as a corollary. With reflection one can develop software engineering tools that examine or produce code. Reflection facilitates testing and problem determination by facilitating the automation of more tedious tasks.

The Java programming language contains a highly effective reflection facility. The tutorial explains the concept of reflection, the Java metaobjects (including both introspective and intercessional interfaces), the proxy class, and dynamic compilation and class loading. The limits of Java reflection are addressed in the context of what reflection is capable of in general. In addition, the tutorial demonstrates the efficacy of the Java reflection facility for solving practical problems. Such problems include: program/application testing, generation of code, inspection of code, and use of dynamic class loading in a framework for application extension. The last topic covered will be the performance impact of using reflection.

Earlier versions of this tutorial inspired the speakers' book "Java Reflection in Action"(Manning 2004), which also provides supplementary material for the tutorial.

20. Introduction to Concurrent Programming in Java 5.0

Monday, 8:30-12:00 ***Pacific Salon 5***

BEGINNER: It is assumed that the attendee is familiar with basic OO concepts and has a working knowledge of the Java programming language.

David Holmes, DLTeCH Pty Ltd

David Holmes is Director and Chief Scientist of DLTeCH Pty Ltd, located in Brisbane, Australia. His work with Java technology has focused on concurrency and synchronization support in the language and virtual machine and he has most recently been working on a real-time Java virtual machine. David was a member of the JCP Expert Group for JSR-166 "Concurrency Utilities", that shipped in the Java 5.0 release. He has presented tutorials on concurrent Java programming and design at numerous international object-oriented programming conferences over the past eight years. Along with Ken Arnold and James Gosling, he is a co-author of the book "The Java Programming Language" - Third and Fourth Editions. David completed his Ph.D. at Macquarie University, Sydney, in 1999, in the area of synchronization within object-oriented systems.

Brian Goetz, Quiotix Corp

Brian Goetz is a Principal Consultant at Quiotix, a software development and consulting firm in Los Altos, California. He has published over 60 articles on Java development in major industry publications. Brian was a member of the JCP Expert Group for JSR-166 "Concurrency Utilities", that shipped in the Java 5.0 release. He is also a current member of the JCP Expert group for JSR 107 "Java Temporary Caching API". He is a co-author of the book "Java Concurrency in Practice", to be published in October 2005 by Addison-Wesley.

Concurrent programming has mostly been the domain of systems programmers rather than application developers, but Java's support for concurrency has enticed many to try their hand at building concurrent applications. Concurrent programming poses many traps for the unwary, however.

This tutorial demonstrates various design patterns and techniques for constructing concurrent applications in Java and for managing that concurrency. On the language side, we examine Java's mechanisms to support concurrent programming, together with some of the more advanced API's from the java.util.concurrent package. On the design side, we explore object structures and design rules that can successfully resolve the competing forces (safety, liveness, efficiency, coordination, reusability) present in concurrent software design problems.

21. Agile Requirements

Monday, 8:30-12:00 ***Sunset***

BEGINNER: This tutorial is appropriate for anyone wanting to specify functional requirements more efficiently, without sacrificing the effectiveness of their communication. Knowledge of common requirements artifacts such as use cases and UML notation is beneficial but not required.

Jennitta Andrea, ClearStream

Jennitta Andrea (jennitta@agilecanada.com) is a partner and senior consultant with ClearStream Consulting since 1994 and has been a practitioner of XP and Scrum on over fifteen different projects since 2000. Jennitta's professional experience spans a variety of roles: agile process coach, requirements analyst, developer, customer quality advocate, instructor, and retrospective facilitator. Jennitta has published and presented at a variety of venues, with an emphasis on process adaptation and automated functional testing. Jennitta is an industrial trainer (2-day Automated Testing course; 3-day Agile Requirements course/workshop), and delivers simulation-based conference tutorials (Agile Requirements; Functional Testing Essentials; Facilitating Effective Project Retrospectives).

Geoff Hardy, Clearstream Consulting

Geoff Hardy is a senior developer with Clearstream Consulting and has been working in the IT industry since 1997. He has experience on a wide variety of projects in Canada and New Zealand, both as a developer and team leader. Geoff has been doing agile development since 2002.

Tailoring the Functional Requirements Specification Process to Improve Agility

Every project has different needs, goals, and constraints, so one size of requirements process does not fit all agile projects. This tutorial explores strategies for tuning the requirements process to optimize agility for a given set of project factors (e.g. team size, project complexity, project criticality, team knowledge and experience, stability and completeness of requirements, etc). Topics discussed include: What work products do we need to use? How much detail is required? How formal does the communication channel, notation, and tools really need to be? Where is the hidden waste?

22. Extending the Standard Widget Toolkit— How to Create Your Own Widgets

Monday, 8:30-12:00 ***Pacific Salon 3***

INTERMEDIATE: Attendees are expected to have a working knowledge of SWT widgets, graphics and layouts.

Veronika Irvine, Eclipse Committer for IBM Canada

Veronika Irvine is one of the original members of the Eclipse SWT team, which she joined in 1998. Her primary areas of responsibility are layouts, drag-and-drop, Active X integration and custom widgets. She has written several articles for eclipse.org, including "ActiveX Support in SWT" and "Adding Drag and Drop to an SWT Application". Veronika works at the IBM OTI Lab in Ottawa.

Steve Northover, Eclipse Committer for IBM Canada

Steve Northover is the principal architect of SWT. He is the SWT team lead for the Eclipse project, and works at the IBM OTI Lab in Ottawa. His areas of expertise include performance, operating system programming and native user interface toolkits. He is the lead author of the book "SWT: The Standard Widget Toolkit, Volume 1".

SWT is a widget toolkit for Java designed to provide efficient, portable access to the user-interface facilities of the operating systems on which it is implemented. It provides a rich set of native widgets for user interface design. However, specialized applications often require widgets that are unique to a particular problem. SWT is extensible in several ways. Complex widgets can be created from the simpler native building blocks; legacy widgets can be encapsulated in a Java layer; or graphics calls and low level user events can be used to design a widget in Java. This tutorial will explore the different SWT extension strategies and show you how to use them to integrate with SWT. Some of the features that will be discussed are advanced graphics, double buffering, custom layout, accessibility, how to successfully extend the SWT hierarchy, and how to maintain a native look and feel.

23. Foundations of object-oriented languages: Types and Language Design

Monday, 8:30-12:00 ***Pacific Salon 2***

ADVANCED: Attendees should know one or more class-based object-oriented programming languages and be experienced with static typing systems.

Kim Bruce, Pomona College

Kim Bruce is Professor of Computer Science at Pomona College. He has taught at Princeton University and Williams College, and has been a visiting professor or scientist at M.I.T., Stanford, Ecole Normale Superieure, University of Pisa, Newton Institute for Mathematical Sciences at Cambridge University, and UC Santa Cruz. He has presented papers at ECOOP, OOPSLA, POPL, MFPS, and LICS, and is the author of Foundations of Object-Oriented Languages: Types and Semantics, published by MIT Press, and Java: An eventful approach, published by Prentice Hall. He received the SIGCSE 2005 award for outstanding contributions to Computer Science Education.

Static typing aids in earlier error detection, supports compiler optimizations, and provides information to programmers on the intended use of constructs. However, simple static typing disciplines for object-oriented languages like C++ and Java are so restrictive that programmers are forced to bypass the type system with type casts. Other languages allow more freedom, but require run-time checking to pick up the type errors that their more permissive systems missed.

After surveying problems with existing type systems, we explain contravariance and covariance in type systems, and suggest ways of improving the expressiveness of these systems while retaining static type safety. Constructs introduced include “F-bounded polymorphism” and “ThisClass”. We include a brief discussion on how the type system and semantics ensure type safety. We compare the strengths and weaknesses of the F-bounded polymorphism corporated into Java 5 with alternative proposals using “where” clauses, match-bounded polymorphism, and virtual types. We also consider new approaches to forming simultaneous specializations of tightly interconnected classes.

24. Use-case patterns and blueprints

Monday, 8:30-12:00 ***Pacific Salon 7***

INTERMEDIATE: Basic knowledge of use-case modeling is needed, and general knowledge of object technology is recommended.

Gunnar Overgaard, Jaczone AB

Gunnar Overgaard has been using, mentoring, and teaching use cases as well as participated in the development of the use-case concept since 1987. He has also participated in the development of UML since 1997. Gunnar has given 100s of classes and presentations, both for industry and academia for up to 150 participants.

Karin Palmkvist, Generic Integration AB

Karin Palmkvist has been working with use cases since the late 1980s, as a system analyst, mentor, teacher, and speaker at conferences and seminars. She also participated in the development and standardization of the UML within the OMG.

Use-case modeling is a well-established technique for capturing requirements stating how a system is to be used. However, many struggle with how to produce complete and correct models and find themselves solving more or less the same modeling problems over and over again. This tutorial presents a collection of use-case patterns and blueprints gathered over a couple of decades, and illustrates how they could be used to facilitate the development of accurate and understandable use-case models.

25. Effective Interface Design

Monday, 8:30-12:00 ***Sunrise***

INTERMEDIATE: This tutorial is targeted at people who write code for a living, typically using a curly-bracket language (C++, C#, Java, etc.) and are looking for some extra advice on how to partition their code with stable and intentional interfaces.

Kevlin Henney, Curbralan Limited

Kevlin Henney is an independent consultant and trainer based in the UK. The focus of his work is in programming languages, OO, UML, agile development, patterns, and software architecture. At various times he has been a regular and irregular columnist for C/C++ Users Journal (online), Application Development Advisor (UK), JavaSpektrum (Germany), Java Report, and C++ Report. He is also on committees and advisory boards for various conferences, language standards, and other software-related organizations (always keeping in mind that “a committee is a cul-de-sac down which ideas are lured and then quietly strangled”...), and is a popular speaker at conferences in North America and Europe.

Seven Recommendations for Improving the Design of Interfaces in Code

Much is made of the pure interface mechanism found in Java, C#, IDLs, and other languages, but this enthusiasm is often not accompanied with advice on effective use. Interfaces are often sold short as a poor relative of abstract classes. How can developers design interfaces that are stable, sufficient, extensible, and sensitive to their context of use, such as callbacks or multithreading?

This talk selects seven recommendations that support effective interface design, whether the interfaces in question are object oriented, procedural, purely abstract, or simply the publicly available methods on a class. The recommendations are drawn from the thirteen recommendations of the Programmer’s Dozen, presented at two previous OOPSLA conferences. C, C++, C#, Java, CORBA IDL, and UML are used to illustrate various examples, counterexamples, and principles.

26. Merciless Refactoring with Eclipse

Monday, 8:30-12:00 ***Pacific Salon 6***

BEGINNER: Attendees must have a solid understanding of object-orientation. Basic understanding of refactoring or Eclipse is not required.

Martin Lippert, it-agile GmbH

Martin Lippert is a consultant and coach for software engineering technology and agile development methods. He focuses on software development on top of the Eclipse platform, refactoring, extreme programming and aspect-oriented programming. He worked for the AspectJ project at PARC as an intern during the summer of '99 and has given a number of talks, tutorials and demonstrations on various topics of software engineering at international conferences. He is co-author of the upcoming book on complex refactorings in large software projects that will be published by Wiley in October.

The goal of the tutorial is to provide an insight how the refactoring support that is built into the Eclipse IDE can be utilized to do merciless refactoring in a safe way. This includes:

- Small refactorings and their usage via keystrokes. This shows how refactorings like "Extract Local Variable", "Convert Local Variable into Field" or "Rename" can be used via simple keystrokes to support even simplest programming tasks.
- Changing method signatures via the proper refactoring feature to automatically adapt all callers.
- Using "Inline Method" to remove deprecated method calls.
- Split up larger refactorings by combined smaller ones. E.g. replace a constructor with a different one by combining "Introduce Factory" and "Inline Method".
- Pushing around classes with drag & drop techniques until a clean package structure is achieved. We demonstrate how this cleanup process can be supported by XRadar, a code report tool. XRadar generates a graphical architecture view depicting architectural violations. By pushing the classes into the appropriate packages the graphical view will show the progress.

Examples for the refactorings are taken from the books by Martin Fowler, Joshua Kerievsky and William Wake.

27. Functional Acceptance Testing: The Essentials

Monday, 13:30-17:00 ***Pacific Salon 7***

BEGINNER: This tutorial is appropriate for anyone that wants to learn how to improve the effectiveness of their functional testing.

Jennitta Andrea, ClearStream

Jennitta Andrea (jennitta@agilecanada.com) is a partner and senior consultant with ClearStream Consulting since 1994 and has been a practitioner of XP and Scrum on over fifteen different projects since 2000. Jennitta's professional experience spans a variety of roles: agile process coach, requirements analyst, developer, customer quality advocate, instructor, and retrospective facilitator. Jennitta has published and presented at a variety of venues, with an emphasis on process adaptation and automated functional testing. Jennitta is an industrial trainer (2-day Automated Testing course; 3-day Agile Requirements course/workshop), and delivers simulation-based conference tutorials (Agile Requirements; Functional Testing Essentials; Facilitating Effective Project Retrospectives).

Functional testing is challenging whether you are working on an agile project that develops tests before developing system code, or working on a more traditional project that has left functional testing until the end. Different choices must be made in each situation in order to develop a successful functional testing strategy. It is crucial that automated functional test suites employ established testing best practices in order to avoid common pitfalls. Topics covered include:

- Context: How functional tests fit into the overall spectrum of testing. Where functional tests fit into the overall project development cycle.
- Strategy: A functional testing strategy is a collection of decisions relating to creating, maintaining, and executing functional tests. Various project characteristics drive these decisions.
- Specification: Functional tests can be textual, graphical, tabular, or storyboarded. Emphasis is placed on developing a domain specific testing language, and other best practices for making your tests a valuable project asset.
- Automation: Several tools are compared with respect to readability, maintainability, declarativeness, effort, results reporting, etc.
- Case-Study: Description of a project that evolved its focus over time (from porting to enhancements), and how the functional testing strategy was developed to evolve successfully.

28. Programming without a Call Stack—Event-Driven Architectures in Action

Monday, 13:30-17:00 ***Sunset***

INTERMEDIATE: This tutorial is targeted at architects and developers who have development experience in a major OO language, e.g. C++, Java, C# etc.

Gregor Hohpe, ThoughtWorks, Inc.

Gregor Hohpe leads the Enterprise Integration practice at ThoughtWorks, Inc., a specialized provider of application development and integration services. Gregor is a widely recognized thought leader on asynchronous messaging architectures and co-author of the seminal book "Enterprise Integration Patterns" (Addison-Wesley, 2004). Gregor speaks regularly at technical conferences around the world and maintains the Web site www.eaipatterns.com.

Most applications today are based on the basic tenets of a call stack-based architecture: one method calls another one, and resumes execution when the called method completes. However, an increasing number of new, exciting technologies are based on asynchronous, event-driven processing models. For example, new libraries in Java J2SE5 introduce the notion of queues and futures, the .Net platform sports delegates, Microsoft's Indigo platform is based on message exchanges, and most industry experts now agree that successful service-oriented architectures are based on message exchange models.

These technologies are powerful and enable loosely coupled communication between processing nodes. But they also move us out our comfort zone of the synchronous, call-stack world. This talk looks behind the specific technologies and examines patterns and best practices to design successful event-driven architectures.

You will learn effective design, debugging and visualization strategies for event-driven architectures. Case studies are presented in both Java and C#, while maintaining the focus on architectural principles over language syntax.

29. Find Your Voice

Monday, 13:30-17:00 ***Pacific Salon 6***

BEGINNER: Participants should have a desire to learn about the intangible factors that affect the success of a project, and to learn how to adjust their personal style in everyday interactions to aid progress and understanding.

Gail E. Harris, Instantiated Software Inc.

Gail has 18 years experience working in the IT field. Her work as a Project Manager, Business Analyst, and Lead Designer for many projects, large and small, has required a finely tuned approach to communicating in a wide variety of situations. In negotiating situations Gail has been able to develop contracts with which all parties found to be not just "satisfactory" but even to their benefit. In design meeting situations Gail has facilitated solutions where every team members is eager to implement the design. Many project team members have said, after the end of a project, that they would like to work with Gail again.

Do you get tongue tied at important meetings? Do you wonder why your ideas are never adopted? Do you wish your colleagues and clients understood you better? If this is happening to you then this is the tutorial for you. In this tutorial we will examine strategies and techniques for communicating in a technology based work environment, regardless of methodology. Starting with one-on-one situations, and working up to the typical mid-size design and requirements meeting groups, you will learn how to analyze the communication situation and your audience. We will explore situations requiring elicitation, negotiation, persuasion, and explanation. You will learn to recognize different styles of communicating, and how to use this knowledge to achieve your goals. Several times during the tutorial we will break up into groups to put into practice what you are learning. Each exercise will include time to get personal feedback on how well you applied this knowledge.

30. Effective Concurrent Programming with Java 5.0

Monday, 13:30-17:00 ***Pacific Salon 3***

INTERMEDIATE: Attendees must be familiar with basic OO concepts and have a working knowledge of the Java programming language. Familiarity with Java's threading and synchronization mechanisms is desirable, but not essential, but attendees should be familiar with the basic principles behind multithreaded programming.

David Holmes, DLTeCH Pty Ltd

David Holmes is Director and Chief Scientist of DLTeCH Pty Ltd, located in Brisbane, Australia. His work with Java technology has focused on concurrency and synchronization support in the language and virtual machine and he has most recently been working on a real-time Java virtual machine. David was a member of the JCP Expert Group for JSR-166 (Concurrency Utilities), that shipped in the Java 5.0 release. He has presented tutorials on concurrent Java programming and design at numerous international object-oriented programming conferences over the past eight years. Along with Ken Arnold and James Gosling, he is a co-author of the book "The Java Programming Language" - Third and Fourth Editions. David completed his Ph.D. at Macquarie University, Sydney, in 1999, in the area of synchronization within object-oriented systems.

Tim Peierls, Prior Artisans LLC

Tim Peierls is President of Prior Artisans, LLC, which he co-founded in 2001. He was previously a Vice President in charge of new technology at Descartes Systems Group, Inc. and Vice President of the Lightstone Group, Inc., which he co-founded in 1989. He was a member of the Expert Groups for JSR 166 (Concurrency Utilities) and JSR 201 (Language Enhancements), that shipped in the Java 5.0 release. Tim is a co-author of the book "Java Concurrency in Practice", to be published in October 2005 by Addison-Wesley. He holds a M.S. in Computer Science from Cornell and a B.S. in Computer Science from Yale.

Brian Goetz, Quiotix Corp

Brian Goetz is a Principal Consultant at Quiotix, a software development and consulting firm in Los Altos, California. He has published over 60 articles on Java development in major industry publications. Brian was a member of the JCP Expert Group for JSR-166 (Concurrency Utilities), that shipped in the Java 5.0 release. He is also a current member of the JCP Expert group for JSR 107 "Java Temporary Caching API". He is a co-author of the book "Java Concurrency in Practice", to be published in October 2005 by Addison-Wesley.

Joe Bowbeer, Mobile App Consulting

Joe Bowbeer is an independent consultant specializing in Java technology in the Java 2 Micro Edition (J2ME) Platform. Joe was a member of the JCP Expert Group for JSR-166 (Concurrency Utilities), that shipped in the Java 5.0 release. He is a contributor to the book "Java Concurrency in Practice", to be published in October 2005 by Addison-Wesley.

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. Designing concurrent programs that can successfully resolve the competing forces of safety, liveness, efficiency, coordination, and reusability, is a task that can be greatly aided through the use of a good concurrency utility library.

The new concurrency utilities in the Java 2 Platform Standard Edition 5.0 release provides a range of tools to allow the programmer to develop concurrent applications more effectively.

This tutorial gives an overview of the Java concurrency utilities and focuses on some of the more sophisticated uses of those utilities. We demonstrate various design patterns and techniques for constructing concurrent applications in Java and for managing that concurrency—with emphasis on the Executor framework and the effective use of Futures. We'll take a trip under the covers of the various synchronizer utilities to see how AbstractQueuedSynchronizer can be used to construct custom synchronization tools. Finally, we look at the advanced thread control API—park/unpark—to see how it can be used to simplify advanced thread coordination problems.

31. STL Patterns: A Design Language of Generic Programming

Monday, 13:30-17:00 ***Pacific Salon 5***

INTERMEDIATE: This tutorial is targeted at C++ programmers with basic knowledge of patterns and some knowledge of the STL.

Kevlin Henney, Curbralan Limited

Kevlin Henney is an independent consultant and trainer based in the UK. The focus of his work is in programming languages, OO, UML, agile development, patterns, and software architecture. At various times he has been a regular and irregular columnist for C/C++ Users Journal (online), Application Development Advisor (UK), JavaSpektrum (Germany), Java Report, and C++ Report. He is also on committees and advisory boards for various conferences, language standards, and other software-related organizations (always keeping in mind that "a committee is a cul-de-sac down which ideas are lured and then quietly strangled"...), and is a popular speaker at conferences in North America and Europe.

Patterns offer an effective way of capturing idioms, allowing programmers a way of communicating and reasoning about their designs through a structured design vocabulary. However, many C++ programmers are trapped in a view of patterns that extends only as far as, or not much further than, the 23 eleven-year old Gang of Four patterns. Useful and seminal as these are, they do not provide the modern C++ developer with a sufficient vocabulary to communicate and reason about their designs.

This tutorial focuses on the generic programming concepts expressed in and around the Standard Template Library, communicating these as C++-specific patterns and connecting them together to form a language that describes the design and use of the STL, illustrating how generic programming is more than simply programming with templates.

32. Domain-Driven Design: Putting the Model to Work

Monday, 13:30-17:00 ***Sunrise***

ADVANCED: Basic object-oriented modeling. Ability to read UML. Complex software development experience helpful. Familiarity with Agile development or Extreme Programming helpful.

Eric Evans, Domain Language, Inc

Eric Evans is a specialist in domain modeling and design in large business systems. Since the early 1990s, he has worked on many projects developing large business systems with objects and has been deeply involved in applying Agile processes on real projects. Out of this range of experiences emerged the synthesis of principles and techniques shared in the book "Domain-Driven Design," Addison-Wesley 2003.
Eric now leads "Domain Language", a consulting group which coaches and trains teams to make their development more productive through effective application of domain modeling and design.

Large information systems need a domain model. Development teams know this, yet they often end up with little more than data schemas which do not deliver on the productivity promises for object design. This tutorial delves into how a team, developers and domain experts together, can engage in progressively deeper exploration of their problem domain while making that understanding tangible as a practical software design. This model is not just a diagram or an analysis artifact. It provides the very foundation of the design, the driving force of analysis, even the basis of the language spoken on the project.

The tutorial will focus on three topics:

- The conscious use of language on the project to refine and communicate models and strengthen the connection with the implementation.
- A subtly different style of refactoring aimed at deepening model insight, in addition to making technical improvements to the code.
- A brief look at strategic design, which is crucial to larger projects. These are the decisions where design and politics often intersect.

The tutorial will include group reading and discussion of selected patterns from the book "Domain-Driven Design," Addison-Wesley 2003, and reenactments of domain modeling scenarios.

33. Large-Scale Software Architecture: A Practical Guide Using UML

Monday, 13:30-17:00 ***Pacific Salon 2***

INTERMEDIATE: Attendees must have a basic understanding of UML, have experience as a software developer, and an interest in software architecture. Experience on a large-scale software system is beneficial but not required.

Jeff Garland, CrystalClear Software, Inc

Jeff Garland has worked on many large-scale software projects over the past 20 years, in many different domains, including telephone switching, industrial process control, satellite ground control, and financial systems. He has served as both the lead architect and a member of the architecture team on several of these projects. Mr. Garland holds a Master's degree in Computer Science from Arizona State University and a Bachelor of Science in Systems Engineering from the University of Arizona. He is currently President and Chief Technology Officer of CrystalClear Software. CrystalClear Software is a consulting firm that specializes in the development of software architectures for large-scale systems.

Richard Anthony, General Dynamics Decision Systems

Richard Anthony has 19 years experience working on large-scale software development efforts. The systems are from application areas such as satellite and network operations systems, telephony base station control, manufacturing, and simulation. He has served in the role of chief software architect, design engineering technical lead, software design lead, software system engineer, and developer on projects in these application areas. Mr. Anthony holds Master's degrees in Computer Science and Mathematics, as well as a Bachelor's degree in Mathematics Education, all from the University of Wyoming. He is currently a Senior Software Architect at General Dynamics Decision Systems.

Dealing with the complexity of large-scale systems can be a challenge for even the most experienced software designers and developers. Large software systems can contain millions of elements, which interact to achieve the system functionality. Managing and representing the complexity involved in the interaction of these elements is a difficult task. This tutorial presents an overview of software architecture terminology and approaches, focusing on a set of UML viewpoints that represent the important aspects of a large-scale software architecture. These viewpoints include context, component, subsystems, process and deployment. These viewpoints leverage the recent IEEE 1471 standard for software architecture representations providing a description of the purpose, stakeholders, and techniques associated with each viewpoint. In addition to introducing the terminology and viewpoints, the tutorial describes other practical techniques essential to developing an effective software architecture. Topics covered in the tutorial include:

- Techniques for handling large complex systems
- Modeling of software subsystem dependencies and interfaces
- Modeling of components, component interactions, and component integration
- Modeling of process communication and software/hardware deployment
- Fitting architecture development into development processes

34. Eclipse Modeling Framework (EMF)

Monday, 13:30-17:00 ***Pacific Salon 4***

INTERMEDIATE: Participants should be familiar with the object-oriented paradigm. Being fluent in one OO language, such as Java, will be useful.

Marcelo Paternostro, IBM

Since 2000, Marcelo has been working for IBM developing products and components targeting the Eclipse platform. In the last 2 years, he has been focused on the EMF, SDO and XSD frameworks, being responsible for components such as the EMF Runtime. He has presented several workshops, both inside and outside IBM, including this year's "MDA, SOA and Web Services Workshop" offered by OMG. Before 2000, Marcelo used to work for a Rational partner, managing several projects using the Rational's tools and processes. As a certified instructor, he has taught several courses on OO analysis and design, requirement management and functional testing.

What it is and how you can use it in your applications.

This tutorial provides an in-depth look at developing with EMF, first highlighting the highest value features of the framework, which every EMF developer should know about, and then delving into the darker corners to expose some of its lesser known capabilities. It also features some of the new and surprising ways in which people have used and reused components of EMF to solve their problems. Concrete examples and demos will motivate the use of EMF and show how to:

- Generate a model from UML, XML Schema, or a set of Java interfaces, with just a few mouse clicks.
- Generate efficient Java code.
- Generate a sample editor that integrates into Eclipse or runs as a Rich Client Platform application.
- Use adapters to respond to model changes and extend behavior.
- Add constraints to model definitions and validate instance models against them.
- Record the changes made to objects
- Customize code generation to control what elements appear in the model interface, or even to remove the reflective EMF API altogether.

The last part of the tutorial describes the motivation behind the Service Data Objects specification, its API and the challenges that architects and developers can overcome by adopting SDO.

WT1. Getting Started with Wikis

Tuesday, 13:30-17:00 ***Royal Palm Salon 3***

BEGINNER: Attendees with either no knowledge of wikis, or those with minimal knowledge of wikis in actual practice will benefit from this tutorial. Attendees who wish to build or improve their own wikis are encouraged to come prepared with an Internet-enabled laptop and an objective for some collaborative knowledge management problem (e.g. a company knowledge base; user manual for an open source project; organizing a neighbourhood civic action).

Sunir Shah, University of Toronto / Socialtext

In April 2000, Sunir Shah founded MeatballWiki, which has grown to be the preeminent place to discuss wiki development and practice. Both Shah and Meatball have always taken a social view of wiki design and information technology in general. Currently Shah is a Masters candidate at the University of Toronto's Faculty of Information Studies and Knowledge Media Design Institute studying social information design. He works for Socialtext as a user-centred design specialist. Shah frequently coaches wiki proprietors on how to manage their wikis.

Wikis may be the hot new buzzword, but they already have 10 years of history. Although often pitched as a simple technology, wikis really are a social practice that uses a simple technology to accomplish a powerful goal. This tutorial introduces wikis, and how they are typically used in actual practice to accomplish collaborative knowledge management. For those new to wikis, both the technology and social practices will be taught, typically in conjuction. Expect to learn the general user experience, how to create a wiki, attract participants, maintain it, and organize groups with it. Audience members will be invited to build their own wiki if they have Internet access during the session.

WT2. The Wikipedia Experience (Cancelled)

Tuesday, 13:30-17:00 ***(Cancelled)***

INTERMEDIATE: Attendees should either be users of large wikis or be responsible for managing one or intend to start one.

Jimmy Wales, Wikimedia Foundation

Jimmy Wales is the creator and driving force behind Wikipedia and the Wikimedia Foundation.

This tutorial has been cancelled due to unforseen circumstances.

This tutorial presents my experiences with seeding, growing, and managing Wikipedia, the online encyclopedia, and its related wikis and endeavors. The talk focusses on the social and managerial aspects of Wikipedia. We discuss practical issues of dealing with wikispam, with editorial processes, decision making within the Wikimedia foundation. We distill our experiences into a set of social best practices so that we hope other wikis can learn from our experiences.

WT3. Using Wikis in Software Development

Tuesday, 13:30-17:00 ***Royal Palm Salon 4***

INTERMEDIATE: Should be software developers or managers.

Jack Bolles, Thoughtworks

Jack: "I've used wikis to varying degrees of success and uptake on most of my projects since 2000. I've seen them be both highly successful, and barely used; both developer-only and project-focused; both organic and mandated. That's a cube of experience rather than binary."

This tutorial shows how to use wikis in software development. We discuss the strengths of using wikis and how they are superior to alternative tooling, in particular with respect to ease of use, integration with other tools, configuration management, agile methods, and opening or limiting access to people out side the team.

We also demonstrate how to integrate wikis with other tools common on a delivery team, including project planning, continuous integration, bug tracking and integration testing (eg. Story Studio, Cruise Control, Jira, and FIT).

Finally, we provide some structure and templates for development, including stories, devevelopment tasks, testing, retrospectives and planning meetings.

We encourage anyone currently using a wiki to share a brief tour of their wikis, if available.

37. Organizational Patterns: Beyond Agility to Effectiveness

Tuesday, 13:30-17:00 ***Royal Palm Salon 1 & 2***

BEGINNER: Attendees should have experience in one or more roles in software development, such as developer, manager or tester.

Neil B. Harrison, Avaya

Neil Harrison is a distinguished member of technical staff at Avaya Labs, where he develops communications software. He has been involved in software development and research for over 20 years, both as a developer and team leader. He has studied software development organizations for ten years. Neil has been a leader in the software patterns community since 1994. He has taught patterns courses and published patterns. He is acknowledged as the world's leading expert on pattern shepherding, and has a shepherding award named after him. He is on the board of directors of the Hillside Group.

James O. Coplien, DAFCA, Inc.

Jim Coplien is a scientist at DAFCA, inc, and a professor of Computer Science at Vrije Universiteit Brussel. His career spans over 20 years at Bell Laboratories, where he directed much of the early industry work on pattern languages in Bell telecommunication projects. Prof. Coplien was a founding member, and is currently Member Emeritus, of the Hillside Group, which launched the software pattern discipline. He continues to do research in formal pattern foundations. His book Organizational Patterns of Agile Software Development, authored jointly with Neil Harrison, reflects a decade of research into organizations world-wide.

It's not just agility. What do some organizations have that enables them to deliver high quality software on schedule, time after time? What is inherent in such organizations that make them consistently effective? We have captured these structural characteristics as patterns of effective software organizations. These patterns contain many of the underpinnings of popular agile methodologies such as XP and SCRUM, but go beyond them to encompass most organizational issues of software.

The patterns cut across all parts of an organization, so we encourage people who fill all roles in software development to attend. It could be particularly interesting for different people within the same organization to attend.

In this tutorial you will learn the many of the most important organizational patterns. You will also learn how they can complement your existing organization improvement practices with these patterns. You will participate in a mock organizational analysis, which will draw from your own experiences in software development. Through this, you will gain some insight into your own organization, and the roles you play. Will this insight inspire you, concern you, or frighten you? It depends on your own organization!

38. PHP/MySQL for Community Programming

Tuesday, 13:30-17:00 ***Royal Palm Salon 6***

INTERMEDIATE: Programming experience in an OO programming language. Interest in software architectures and development methodologies. Familiarity with HTML is helpful.

Ghica van Emde Boas, Bronstee.com Software & Services

Ghica van Emde Boas is an independent consultant. After a career of 30 years as Software Architect and Java developer with IBM, she spent 4 years as consultant and designer for the IBM SanFrancisco framework (the largest Java framework ever developed), mostly in Sweden and the Netherlands. Since early 2004 Ghica is doing community projects like the ones mentioned above, using PHP/MySQL. She is project leader (with Jorn Bettin) of the Generative Model Transformer open source project (www.eclipse.org/gmt). Ghica is co-organizer of the OOPSLA workshops on Model-Driven Software Development, expected to be held for the 4th time this year.

PHP is a web-scripting language, primarily used in emerging countries. Community programming is: developing a bulletin board for your grandmother with Alzheimer, web-publishing facilities for your local sports club, or a small shop.

Why should you care?

PHP is open source, easy to learn, productive, easy to deploy. PHP5 has full object-oriented features. PHP has extensive web-functionality built-in, including a strong MySQL interface.

Businesses will find that well defined web-architectures, and good development methods for PHP are still lacking. Instead of looking at PHP, they will consider outsourcing the development of their web-applications... unless you can combine your existing knowledge of development methods with new skills of highly productive PHP coding.

This tutorial has a more modest goal than presenting enterprise development methods for web applications in PHP. The first half will be a quick introduction to PHP. During the second half will we look at an open source PHP Application Framework aimed at community applications and using model-driven generative techniques.

We want to spark your interest in PHP. The rest has to be done by you. And we hope you make a difference to your community!

39. First-Aid Clinic for Change Agents

Tuesday, 13:30-17:00 ***Stratford***

BEGINNER: Anyone who is trying to introduce new ideas and is at least familiar with the notion of patterns.

Linda Rising, Independent consultant

Linda Rising has a Ph.D. from Arizona State University. She has been working with object technologies since 1983. She is the editor of A Patterns Handbook, The Pattern Almanac 2000, and Design Patterns in Communication Systems. She has a number of publications including: "Patterns for Collaboration," Cutter IT Journal, February 2005. This and other articles are available on her web site: www.lindarising.org. She has presented a number of tutorials and workshops at JAOO, OOPSLA, and other conferences. Her new book co-authored with Mary Lynn Manns is titled: Fearless Change: Patterns for Introducing New Ideas. She has over 20 years of academic teaching experience and over 15 years of industrial training experience.

Mary Lynn Manns, University of North Carolina at Asheville

Mary Lynn Manns earned her PhD from De Montfort University in England in the area of introducing software patterns into organizations. Her 20-year teaching experience in academia and industry includes an outstanding teaching award in 1995. She is currently at the University of North Carolina at Asheville where she has taught courses in computer science and management information systems. Her new book is now available: Fearless Change: Patterns for Introducing New Ideas, co-authored with Linda Rising. Mary Lynn has done numerous presentations on the topic of introducing new ideas into organizations.

Struggling to help your team or organization become more innovative? Have great ideas but can't seem to get them off the ground? This tutorial is for you! Many people who attend OOPSLA uncover new ideas they want to take back, but then struggle to make something happen. In this tutorial you will hear the secrets of successful change agents as documented in the presenter's latest book: Fearless Change: Patterns for Introducing New Ideas. The tutorial format allows you to learn some of these patterns and apply them to your problems.

40. Agile Software Development in the Large

Wednesday, 13:30-17:00 ***Royal Palm Salon 1 & 2***

INTERMEDIATE
- Change agents and promoters of agile methods
- Executives
- Project managers, product managers, development team managers

It would be helpful if the audience would be familiar with a specific agile process, however it's not a requirement.

Jutta Eckstein, IT communication

Jutta Eckstein (www.jeckstein.com, info@jeckstein.com) is an independent consultant and trainer for over ten years. She has a unique experience in applying agile processes within medium-sized to large mission-critical projects. This is also the topic of her book Agile Software Development in the Large. Besides engineering software she has been designing and teaching OT courses in industry. Having completed a course of teacher training and led many 'train the trainer' programs in industry, she focuses also on techniques which help teach OT and is a main lead in the pedagogical patterns project. She has presented work in her main areas at ACCU (UK), OOPSLA (USA), OT (UK), XP (Italy and Germany) and XP and Agile Universe (USA).

A lot of people still believe that agile software development is for small teams only. However, the agile value system and the principles behind as stated in the agile manifesto don't say anything about team or project size. Furthermore the projects I'm working on are typically large and mission-critical. Therefore, several years ago I took the challenge and tried agile software development in the large. Meanwhile I made the similar experience on many large projects: Also large teams can benefit from a value system that is beneficial for small teams. In this tutorial I want to show how to scale agile processes to teams of 200. In fact, the same techniques are also relevant to teams of ten or more developers, especially within large organizations.

Topics include:

- the agile value system as used in large teams
- the impact of a switch to agile processes
- the agile coordination of several subteams
- the way project size and team size influence the underlying architecture

41. DocIt!: Agile Documentation of Object-oriented Frameworks

Wednesday, 13:30-17:00 ***Royal Palm Salon 3***

INTERMEDIATE: Anyone engaged with framework-based software development.

Ademar Aguiar, Faculdade de Engenharia da Universidade do Porto (FEUP)

Ademar Aguiar teaches at Faculty of Engineering, University of Porto (FEUP) and does Research & Development at INESC Porto, since 1989. He has specialized in the area of software engineering, mainly OO, UML, software architecture, OO frameworks, design patterns, agile processes, and software documentation, topics about which he has authored several research papers and presented many different courses to academic and industrial audiences. His PhD thesis, titled "A Minimalist Approach to Framework Documentation", proposes a pragmatic approach to document frameworks, which comprises a documentation model, an agile documentation process, and a set of collaborative documentation tools and utilities based on the Wiki concept and XML technologies. Currently, his main line of research is on Wiki-based tools to support the low-cost production of high-quality documentation for OO frameworks.

OO frameworks are a powerful technique for large-scale reuse, but they are particularly hard to understand by new users, especially if not accompanied with good documentation, which is typically hard, costly and tiresome to produce without appropriate tools and methods. This tutorial introduces and lets you practice DocIt!, a simple and economical agile approach to produce lean and mean framework documentation, easy-to-adopt even in development environments restrictive to documentation activities. DocIt! comprises a model, an agile process and a set of open tools capable of preserving the semantic consistency between source code, models, and textual contents, all integrated in a Wiki engine and Eclipse. The tutorial drives you to get-started-fast using DocIt! and to learn-by-doing, by helping you on: meeting the needs of different audiences; cooperatively writing typical framework documents, namely overviews, examples, cookbooks, recipes, and pattern instantiations; combining and linking different kinds of contents; and organizing the contents as a minimalist instruction manual, easier to understand and use. You are invited to bring your laptop and your own framework, or to pair with someone who does, or to document the popular JUnit framework.

42. Robust Communications Software

Wednesday, 13:30-17:00 ***Royal Palm Salon 4***

INTERMEDIATE: Participants should have a general familiarity with object orientation. The tutorial will be of particular interest to people who work on, or provide platforms for, communications products such as routers, switches, servers, gateways, and mobile handsets.

Greg Utas, Pentennea LLC

Greg Utas is a consultant specializing in carrier-grade software and the author of the recently published "Robust Communications Software" (Wiley, 2005). He has over 20 years of industrial experience, including Chief Software Architect roles at both Sonim Technologies (wireless push-to-talk services) and Nortel (GSM core networks). He became the first software architect at Nortel's Director level after leading a team of 50 designers who rewrote GSM call handling software using OO techniques.

Carrier-grade software is required in embedded systems that implement communications networks, such as routers, switches, servers, gateways, and mobile handsets. Firms that operate communications networks are known as carriers. A carrier-grade product is one that satisfies a carrier's strict quality requirements, which typically include

- Availability: <5 minutes downtime per year (five nines, or 99.999%)
- Reliability: <1 failure per 10,000 sessions (four nines, or 99.99%)
- Scalability: >100,000 users per system (site); millions of users per network

Although many products claim to be carrier grade, the fine print often reveals that this only refers to their hardware. But in today's complex communications products, software is usually the cause of outages and incorrect behavior. This course covers a broad range of carrier-grade software techniques that have been proven in flagship products and that significantly improve a product's chances of satisfying the most extreme requirements. Many of these techniques are seldom used elsewhere in the computing industry and sometimes supplant common practices that are ill advised in carrier-grade systems. The tutorial includes OO-related techniques that address concerns involving object management, execution overhead, and exception handling.

43. Creating and Protecting Software Intellectual Property Rights

Wednesday, 13:30-17:00 ***Royal Palm Salon 5***

BEGINNER: Anyone interested in creating and maintaining software intellectual property rights.

Dion Messer, Wilson Sonsini Goodrich and Rosati

Ms. Messer is an intellectual property attorney licensed in Texas, California, and at the PTO. Education: JD, University of Texas School of Law; MS Electrical Engineering, University of Texas; and BS Electrical Engineering, New Mexico State University. She is currently a clerk to the Honorable William Bryson on the Court of Appeals for the Federal Circuit. Prior to practicing law, she was an active co-owner of Objective Engineering, Inc., a company that provides training and consulting in object technology, and an engineer for 18 years. She has nine patents and has published extensively in the area of digital signal processing.*
**Pending the formal swearing in scheduled on April 23, 2005.*

In this age of growing intellectual property rights for software and business processes, it is imperative that software developers understand what those rights are and the mechanisms used to protect them. Failing to adequately protect software may leave its owners and developers with no recourse should a third party misappropriate the design or implementation.

This tutorial explains how software intellectual property rights can be protected through patents, copyrights, and trade secrets. It compares and contrasts the procurement, cost, and scope of protection of all three. It also addresses protecting software from both the outside—theft by a third party—as well as from the inside—theft by transient employees, contractors, etc.

Software intellectual property rights are further complicated by the fact that three sets of ownership rules govern contractors, regular employees hired to develop the software, and other employees. This tutorial explains the differences in those rules for each type of protection and their implications to both organizations and individuals.

Finally, employers can implement procedures to protect them from potential lawsuits when they hire ex-employees of competitors. This tutorial explains these procedures, which include stringent software code control to archive each version of the software as it is developed.

44. Making RUP Agile

Wednesday, 13:30-17:00 ***Esquire***

INTERMEDIATE: This tutorial is aimed at project managers, software architects, software process specialists and software developers who are evaluating RUP for agile development or who are involved in a project where RUP is already used and want to make it more agile. Knowledge of RUP basics is helpful but not required.

Michael Hirsch, Zühlke Engineering AG

Michael Hirsch has 20 years of experience in the software industry in various roles, including project manager, software architect and software developer. During the last 12 years, he has been with Zühlke Engineering AG, a software contractor and software consultancy in Switzerland. He has been using RUP since 1998, when he led a team that introduced RUP at Zühlke Engineering and adapted it to the company's needs. Since then Zühlke Engineering has successfully completed about 20 projects with an agile version of RUP. Today, Michael splits his time between managing software development projects, coaching and mentoring project teams, and teaching classes on software processes and object oriented analysis and design. He is a member of ACM and IEEE, and has a degree in electrical engineering from HTL Bregenz in Austria and a degree in software engineering from HTL Berne in Switzerland.

The Rational Unified Process (RUP) is a comprehensive process covering almost all aspects of software development projects. Due to its great level of detail, RUP has - quite wrongly - the reputation of being too heavyweight for agile development projects. In this tutorial, you will learn how to apply RUP for agile development. Topics covered include what artifacts to use and not to use, best practices for planning and monitoring projects, best practices for handling requirements, analysis and design, and how to introduce agile RUP into a project or in an organization. About 25% of the time of the tutorial is devoted to a demonstration of a real world project which has been successfully completed with an agile version of RUP.

45. The Eclipse Debug Framework

Wednesday, 13:30-17:00 ***Royal Palm Salon 6***

INTERMEDIATE: Using the Eclipse IDE and interested either in building a debugger with Eclipse or in the details of how to do so.

Bjorn Freeman-Benson, Eclipse Foundation

Bjorn is the Technical Director of Open Source Process and Infrastructure for the Eclipse Foundation and a Fellow with Bedarra Research Labs. Throughout his meandering career, Bjorn has always been interested programming languages and IDEs in spite of occassionally wandering away from their one true path. Bjorn has worked for OTI, Amazon.com, Rational, and Gemstone, among others. He is also happy to expound at length about his love of flying.

Darin Wright, IBM Rational Software

Darin is currently a senior software developer with the IBM Rational Software Group, an Eclipse committer, and lead for the Eclipse Debug Platform and Java Debugger. For the better part of the last ten years, Darin has been working on IDE's such as Eclipse, VA/Micro Edition, and ENVY/Smalltalk. In a previous software development life, Darin was an audio software developer supporting virtual reality productions at the Banff Centre for the Arts.

This tutorial covers the the design and extension points of the Eclipse Debug framework, and specifically how to add debugger support for a new language or application to the Eclipse IDE. Most of the tutorial examples use a small assembly language for educational simplicity, but the lessons and techniques apply to languages and applications of all complexities. We investigate larger applications of the Debug Framework by showing a few of the details of a few other Eclipse debuggers including: the Eclipse JDT debugger and a debugger for a GEF-based visual language. This tutorial is based on our upcoming Addison-Wesley book "Writing Integrated Debuggers with Eclipse".

The tutorial is divided into a dozen modules. Each module consists of lecture followed by an in-class exercise. The tutorial notes contain much more material than can be covered in a half-day, so after the background modules we ask the audience to choose the issues the rest of the course will cover.

53. An Introduction to Requirements Engineering

Thursday, 8:30-17:00 ***Royal Palm Salon 3***

BEGINNER: No prior knowledge of requirements engineering processes or software development is required. Any business or software professional will be able to follow the material.

Brian Berenbach, Siemens Corporate Research

Presenter: Brian Berenbach is the program manager for requirements engineering at Siemens Corporate Technology. He has been working in the field of requirements engineering for over 15 years, first as a consultant, and then as a senior member of the technical staff at Siemens Corporate Research in Princeton. Recently at Siemens his program has been involved with requirements definition for such diverse products as medical systems, baggage handling, mail sorting, automated warehouses, and embedded automotive systems.

Requirements elicitation and management has become ever more important as products become more complex and time to market is shortened. Outsourcing has added a new dimension to requirements management, exacerbating problems associated with transitioning from analysis to design. This one day survey course will provide an introduction to requirements engineering from the perspective of project and product management: how it impacts project managers, quality assurance personnel, requirements analysts, developers and testers. Topics covered will include product line requirements, feature modeling, CMMI compliant requirements management and requirements analysis processes (both UML and text based). Business analysts who are interested in using UML for modeling will also find the course interesting. No formal knowledge of programming is required.

46. Models and Aspects: How to use MDSD and AOSD together

Thursday, 8:30-12:00 ***Royal Palm Salon 5***

INTERMEDIATE: Attendees must have a solid understanding of object-orientation. Basic understanding of MDSD and AOSD concepts are useful.

Markus Voelter, Independent Consultant

Markus Voelter works as an independent consultant, coach and trainer for software technology and engineering. He focuses on software architecture, middleware as well as model-driven software development. Over the last years, Markus has worked on several model-driven software development projects in the enterprise and embedded world. Examples include banking, automotive and radio astronomy. Markus is a regular speaker at the relevant national and international conferences. For example, he has presented at ECOOP, OOP, OOPSLA, ACCU. Markus is the (co-)author of several patterns, many magazine articles as well as several books on middleware, model-driven software development and software architecture.

Martin Lippert, it-agile GmbH

Martin Lippert is a consultant and coach for software engineering technology and agile development methods. He focuses on software development on top of the Eclipse platform, refactoring, extreme programming and aspect-oriented programming. He worked for the AspectJ project at PARC as an intern during the summer of '99 and has given a number of talks, tutorials and demonstrations on various topics of software engineering at international conferences. For example, he has presented a technical article on error detection and handling using Aspect-Oriented Programming at ISCE and has given demonstrations of an AspectJ-enabled Eclipse Runtime technology he is working on at OOPSLA and AOSD.

Aspect Oriented Software Development (AOSD) as well as Model-Driven Software Development (MSDD) are both becoming more and more important in modern software engineering. Both approaches attack important problems of traditional software development. AOSD addresses the modularization (and thus, reuse) of cross-cutting concerns (CCC). MDSD allows developers to express structures and algorithms in a more problem-domain oriented language, and automates many of the tedious aspects of software development. But how do the two approaches relate? And how, if at all, can they be used together? This tutorial looks at both of these questions. The first one - how AOSD and MDSD relate - is discussed in the first part of the tutorial. The second, and main part of the tutorial introduces six patterns of how MDSD and AOSD can be used together. These range from using code generation templates as a simple means to separate concerns to using aspect-oriented modeling concepts to separate concerns in the models. All the patterns are illustrated with practical real-world examples taken from various model-driven software development projects.

47. Challenges in Object-Relational Mapping

Thursday, 8:30-12:00 ***Royal Palm Salon 6***

INTERMEDIATE: Basic knowledge of object-oriented programming and relational databases is helpful, but not required.

Alan Knight, Cincom Systems

Alan Knight works on Smalltalk database and web tools for Cincom Systems, and is lead on the GLORP open-source object-relational mapping layer. Prior to joining Cincom he was chief architect for TOPLink with The Object People (TOPLink is now owned by Oracle). He was a member of the Sun expert groups on EJB 2.0 and JDO. Alan has spoken extensively at conferences including OOPSLA, Smalltalk Solutions, JAOO, OOP and Java One, and is co-author of the book, Mastering ENVY/Developer. He can be reached as knight@acm.org.

Object-oriented programming and relational databases are two of the dominant programming models of our time. Why is it so difficult to make them work together?

This tutorial examines the core issues, design trade-offs and practical difficulties in using objects and relational databases together. At the architectural level, this affects how queries are defined, how objects are associated with transactions, and concurrency strategies. At the mapping level, we need to consider the mapping of different types of relationships, including inheritance. The difference between programming language nil/null and database NULL is significant, though often overlooked. At all levels, performance, including the ability to performance-tune at either the object or the database level, is critical.

For the most part, the discussion will be language-neutral, but when specific language features are relevant to the mapping, specific examples will be used, drawn primarily from Java and Smalltalk.

48. Building Service-Oriented Architectures with Web Services

Thursday, 8:30-12:00 ***Royal Palm Salon 1 & 2***

BEGINNER: The tutorial is targeted at practitioners with general background in software engineering and object-oriented programming; familiarity with XML and Java/C# is a plus.

Olaf Zimmermann, IBM Emerging Technologies

Olaf Zimmermann is a Senior Certified IT Architect in IBM's worldwide Emerging Technologies jumpStart (jStart) team. His areas of expertise include distributed computing and service-oriented architectures in general and J2EE/Web services in particular. Over recent years, Olaf has conducted numerous Web services-related engagements, and educated practitioners around the world on this technology. He is an author of the Springer text book "Perspectives on Web Services" (ISBN 3-540-00914-0). Olaf also contributed to several IBM ITSO Redbooks such as "Web Services Wizardry with WebSphere Studio Application Developer", SG24-6292-00. Olaf holds an honours degree in Computer Science from the Technical University in Braunschweig, Germany.

Mark Tomlinson, IBM Software Group

Mark Tomlinson is an IBM Consulting IT Specialist, and is based in London, UK. Mark has been working with various Java and XML technologies for the past seven years, including WebSphere since its very first release. He was an author of the IBM ITSO Redbook "Web Services Wizardry with WebSphere Studio Application Developer" and his second book on the subject, entitled "Perspectives On Web Services" (ISBN 3-540-00914-0) was published in July 2003. Mark currently spends his time working with IBM's major customers in the Financial Services sector helping them to adopt new J2EE technologies and move towards service-oriented architectures. He holds a degree in Physics from the University of Warwick, UK.

Service-Oriented Architecture (SOA) and Web Services (WS) technologies have matured into highly attractive architecture and implementation alternatives for building distributed systems. SOA concepts and Web services standards-based implementation stacks are a powerful combination that is well-suited for crafting heterogeneous B2B and EAI solutions.

In this tutorial, we introduce the fundamental concepts that define SOA as a state-of-the-art approach for enterprise application development and integration, and investigate key SOA patterns such as Enterprise Service Bus (ESB) and Business Process Choreography. Next, we explain the core stack of Web services specifications - for example, the SOAP message exchange format, the WSDL interface description language, and the UDDI service broker model. In a third module, we design and develop a complete sample application applying these concepts and technologies, making use of several code generators and runtime environments such as Apache Axis. We conclude with a discussion of the most important architectural decisions to be taken on SOA and Web services development projects - for example WSDL creation process and service interface granularity; SOAP communication style and best practices for interoperability; security, compression and other quality-of-service factors; server-side deployment and client-side invocation guidelines.

49. Agile User Experience Design

Thursday, 8:30-12:00 ***Royal Palm Salon 4***

INTERMEDIATE: Those experienced with Agile methods who wish to understand how UX might be incorporated. Those experienced with UX who have questions on how user experience practices and techniques work in an Agile environment. These two groups represent a broad range of people involved in software development including: development managers, project managers, team leaders, coaches, developers, UI designers, and other user experience practitioners. The material is simple enough to address practitioners at a beginner level, but original enough to be valuable to experienced practitioners.

Jeff Patton, ThoughtWorks Inc.

Jeff Patton has designed and developed software for the past 10 years on a wide variety of projects from on-line aircraft parts ordering to rules-based pricing engines. Since working on an XP team in 2000, Jeff has been heavily involved in Agile methods. In particular Jeff has focused on the application of user experience techniques on Agile projects resulting in leaner, more collaborative forms of traditional UX practices. Jeff has found that adding UX thinking to Agile approaches of incremental development and story card writing not only makes those tasks easier but results in much higher quality software.

User Experience is the blanket term that incorporates user centered design, interaction design, user interface design, and usability. As Agile Development processes gain popularity, the need to include User Experience practitioners becomes more urgent. Many companies employ people in these roles and want advice on incorporating them into their Agile processes. In contrast, many Agile projects not incorporating UX approaches suffer from delivery of products that, though on-time and of high technical quality, poorly address user needs.

In this tutorial you'll learn the basics of the Agile Development and User Experience lifecycles and how User Experience work can function cohesively within Agile Development. You'll learn that UX refers to much more than user interface design and, in fact, encompasses a number of techniques that give guidance on project scoping and user interface validation. Through participation you'll learn lightweight collaborative variations of UX techniques such as user role modeling, task modeling, and user interface prototyping. Using these models and UI prototypes you'll write user stories that incrementally release your product design into the Agile Development lifecycle.

50. The C# Programming Language

Thursday, 13:30-17:00 ***Royal Palm 4***

BEGINNER: Prerequisites: Participants must be familiar with object-oriented programming concepts. Familiarity with Microsoft .Net, C# is useful but not essential.

Mads Torgersen, Microsoft Corporation

Mads Torgersen has recently joined the C# development team at Microsoft.

C# is the object-oriented cornerstone for development in Microsoft's .NET architecture. It's easy to learn C# itself, especially if you're familiar with Java, C++, or other OO languages. This tutorial will introduce the syntax of C#, exposing the design philosophies behind the language, and will show how C# nestles within the .NET Framework. We'll see an overview of the .NET Framework Class Library and the benefits of the Common Language Runtime (including garbage collection). Finally, we'll take a look at what's coming in C# 2.0.

51. An Architects Guide to Enterprise Integration with J2EE and .NET

Thursday, 13:30-17:00 ***Royal Palm 1 & 2***

INTERMEDIATE: You need a solid understanding of object-orientation. You should have an appreciation of the key features of distributed component technologies and enterprise platforms (J2EE, .NET, Web Services).

Ian Gorton, National ICT Australia

Ian Gorton is a Senior Researcher at National ICT Australia. Until Match 2004 he was Chief Architect in Information Sciences and Engineering at the US Department of Energy's Pacific Northwest National Laboratory. Previously he has worked at Microsoft and IBM, as well as in other research positions. His interests include software architectures, particularly those for large-scale, high-performance information systems that use commercial off-the-shelf (COTS) middleware technologies. He received a PhD in Computer Science from Sheffield Hallam University.

Anna Liu, Microsoft

Anna Liu is an architect advisor at Microsoft Australia. She evangelizes architecture and works with large enterprises in understanding and solving their enterprise application integration challenges. Her interests include designing and implementing robust service oriented architectures, and distilling patterns and best practices. She has previously held a visiting scientist position at the Software Engineering Institute, Carnegie Mellon University. She received a PhD in Computer Engineering from the University of New South Wales, Australia.

Architects are faced with the problem of building enterprise scale information systems, with streamlined, automated business processes and web-enabled business functions, all across multiple legacy applications. The underlying architectures for such systems are embodied in a range of diverse products known as Enterprise Application Integration (EAI) technologies. In this tutorial, we highlight some of the major problems, approaches and issues in designing integration architectures and selecting appropriate supporting technology. An architect's perspective on designing large-scale integrated applications is taken, and we discuss requirements elicitation, architecture patterns, EAI technology and features, and risk mitigation. A range of J2EE and .NET technologies, including the Java Messaging Service, Java Connector Architecture, MSMQ, BizTalk and Web Services, are used to illustrate the capabilities of state-of-the-art integration components, and their various strengths and weaknesses are discussed. A case study of a global insurance business is used throughout the tutorial to demonstrate the concepts and technologies that are covered. The case study incorprates data, messaging and service-based integration architectures, and shows how J2EE and .NET technologies can be used to construct a flexible and scalable integration solution.

52. Pattern Languages Hands-On

Thursday, 13:30-17:00 ***Royal Palm Salon 5***

INTERMEDIATE: Attendees should have a familiarity with basic notions of programming.

Maria Kavanagh, De Montfort University

Maria Kavanagh is a Senior Lecturer in the Faculty of Computing Sciences and Engineering at De Montfort University and a member of Hillside (Europe). She has recently completed a doctoral study on pattern languages and is the author of the business process pattern language: APPLE. She has submitted papers and presented workshops at EuroPLoP, OOPSLA, rOOts (Norway) and the ACCU conference in England.

Alan O'Callaghan, De Montfort University

Alan O'Callaghan is a Senior Lecturer in the Faculty of Computing Sciences and Engineering at De Montfort University. He has authored two books on object technology,and more than forty articles and journal papers on OO, software architecture and patterns. He has authored two pattern languages ADAPTOR for the migration of legacy systems, and the Bots n Pieces for building Mindstorms robots. He is a member of the Hillsifde (Europe) Advisory Board.

Patterns and refactorings have been described as being to systems development what gardening is to gardens. In this tutorial attendees are provided with the Bots n Pieces pattern language. This is a language that was designed over two years in long-running experiment, started at EuroPLoP '03, to explore Christopher Alexander's ideas of 'centers' and 'sequences' in pattern languages and to test their relevance to software-intensive systems. Working in groups they refactor a programmable Lego Mindstorms robot (altering both the 'hardware' and the software) in a hands-on exploration of how pattern languages for software-intensive systems can help create 'living structure'. The activities will be recorded on digital video which will form part of the output of the tutorial (together with posters for the rest of OOPSLA).

Wiki Symposium — Sunday/Monday

Chair: Dirk Riehle, Founder and Chief Scientist, Bayave Software

The 2005 International Symposium on Wikis brings together wiki researchers, implementers, and users for the first time. The goal of the symposium is to find a voice for the community. The symposium has a rigorously reviewed research paper track as well as plenty of space for practitioner reports, demonstrations, and discussions. We are honored to announce that Ward Cunningham, the inventor and host of the original WikiWikiWeb, will present the opening keynote talk at WikiSym 2005. He will be followed by Jimmy Wales, founder of Wikipedia, speaking on Wikipedia in the free culture revolution. Anyone who is involved in using, researching, or developing wikis is invited to WikiSym 2005!

Registration for WikiSym gives you access to the full WikiSym program, including workshops, if you have been invited by the workshop organizers. If you want to attend a WikiSym (or OOPSLA) tutorial, you will need to register separately for the tutorial through the OOPSLA website.

Workshop 1: WikiSpam Workshop — Sunday, 13:30-17:00 — *Eaton*

Sunir Shah, *University of Toronto*

Wiki spam has begun to cripple the use of public wikis. The original WikiWikiWeb threw its 'shields up' at the beginning of 2005 in part due to spammers. The problem of defending against wiki spam while preserving the traditional wiki value of openness is a daunting one. MeatballWiki has been leading the discussion of how to defend sensibly against wiki spam. This workshop is intended as both a report on progess as well as a facilitated time to come together to devise better solutions for this problem.

Opening Keynote — Monday, 08:30-10:00 — *California Room*

The Crucible of Cooperation

Ward Cunningham, *Microsoft*

Research Papers 1 — Monday, 10:30-12:00 — *California Room*

Session Chair: Matthias Jugel

Panel: The Future of Wikis — Monday, 10:30-12:00 — *Terrace Pavilion & Poolside*

Moderator: Sunir Shah, *University of Toronto*

Ward Cunningham, *Microsoft*

Jimmy Wales, *Wikimedia Foundation*

Ross Mayfield, *Socialtext*

Research Papers 2 — Monday, 13:30-15:00 — *California Room*

Session Chair: Seb Paquet

Panel: Wikis in the Consumer Enterprise — Monday, 13:30-15:00 — *Terrace Pavilion & Poolside*

Moderator: Eugene Eric Kim, *Blue Oxen Associates*

Joe Kraus, *JotSpot*

Peter Thoeny, *TWiki*

Thomas Weigert, *Motorola Global Software Group*

Enterprise software is changing before our eyes: from the way it is sold, bought and used. The model of top down, million dollar license deals with multi-million dollar service contracts sold over six to nine month sales cycles is quickly becoming a dinosaur. The new model is the "consumer enterprise" where software becomes a commodity: it is inexpensive, deployed from bottom up, lightweight, and adaptable to changing business processes. Among other reasons, wikis are gaining popularity in the enterprise because they fit into this major transition. This panel will explore wikis in the enterprise - how they're being used, who benefits and where it's heading - and the intersection of wikis with the major trend of the consumer enterprise.

Speed Demos — Monday, 15:30-17:30 — *California Room*

Session Chair: Sunir Shah

Research Papers 3 — Monday, 15:30-17:00 — *Ascot Room*

Session Chair: Alain Desilets

WikiSym Reception — Monday, 17:00-19:00 — *Trelisses' Rest. Poolside Area*

Mingle with your peers and other folks and talk about what's happening and what to do next.

OOPSLA 2005 Keynote **Tuesday, 8:30-10:00** ***Town & Country Room***

Creativity

Robert Hass, *Chancellor of The Academy of American Poets*

Wiki Symposium attendees are invited to attend the OOPSLA Keynote presentation, which is described on page 12 in the Invited Speakers section.

Joint OOPSLA/WikiSym Invited Talk **Tuesday, 10:30-12:00** ***Golden West Room***

Wikipedia in the Free Culture Revolution

Jimmy Wales, *Wikimedia Foundation*

Please see the description on page 12 in the Invited Speakers section.

Workshop 2: Wiki-Based Software Documentation **Tuesday, 13:30-17:00** ***Esquire***

Ademar Aguiar, *University of Porto (FEUP)* Gabriel David, *University of Porto (FEUP)*

Wiki's proved to be very appealing collaboration tools to present and edit web-based information, through a very simple markup language, a powerful dynamic-linking mechanism, and support to the notion of adaptive web pages. This workshop focuses on the usage of Wiki engines in the specific domain of software documentation. It aims at bringing together researchers and practitioners interested in researching and exploring Wiki's as cost-effective tools to support cooperative and agile software documentation, with the goal of identifying key opportunities, challenges and obstacles to Wiki-based software documentation, and to build collaborations for future research.

Workshop 3: Open-Source Software Development with Wikis **Tuesday, 13:30-17:00** ***Dover***

Peter Theony Colas Nahaboo

Some open source communities started to use a wiki to organize their work. This workshop uses the TWiki project as a case study. The TWiki community is using their structured wiki since inception as the primary communication vehicle to brainstorm on ideas, track new features, track bugs, provide support and create documentation. Other open source communities using Wikis are invited to participate in the workshop to compare the emerging methodologies of wikis for open source software development.

Workshop 4: Interwiki Workshop **Tuesday, 13:30-17:00** ***Royal Palm Salon 5***

Lion Kimbro Helmut Leitner Bayle Shanks

This workshop will be an informal discussion of ideas and technologies for connecting wikis to each other. Topics may include: (1) Linking between wikis (examples: InterWiki link prefixes, NearLinking, SisterSites, LocalNames) (2) Inter-wiki page interchange (3) RecentChanges syndication (example: RecentNearChanges) (4) Distributed wiki engines (5) Wiki markup standard(s) (6) Interoperability between wiki and other software (7) Software projects for wiki interoperability (examples: OneBigSoup, WikiGateway) (8) Standards relevant to wikis (examples: ModWiki/RSS, Atom, WikiXmlRpc) (9) Social and information architecture initiatives to connect wikis (examples: TourBus, WikiNode) (10) Social standards. The actual topics discussed will depend on what participants want to discuss.

Wiki Tutorials **Tuesday, 13:30-17:00**

If you want to attend a WikiSym (or OOPSLA) tutorial, you will need to register separately for the tutorial through the OOPSLA website.

WT1. Getting Started with Wikis

Sunir Shah, *University of Toronto / Socialtext*

Please see the description on page 77 in the Tutorials section.

WT2. The Wikipedia Experience (Cancelled)

Jimmy Wales, *Wikimedia Foundation*

This tutorial has been cancelled due to unforseen circumstances.

WT3. Using Wikis in Software Development

Jack Bolles, *Thoughtworks*

Please see the description on page 78 in the Tutorials section.

Workshops — Sunday

Chair: Bill Opdyke, North Central College

OOPSLA workshops are fun! These workshops are highly interactive events, where groups of object technologists meet to surface, discuss, and attempt to solve challenging problems. You learn a lot when you share your ideas and experiences with others in the field, and build relationships that are an essential part of the OOPSLA experience. The workshop topics (listed below) are diverse. If you'd like to attend one of them, and haven't already done so, please contact the relevant organizers (click on the link below corresponding to their workshop; their URL is included with the workshop description) as they are responsible for managing attendance. In most cases you will be asked to submit a (short) position paper in advance to ensure that the workshop is appropriate for you. We hope to see you in a workshop at this year's OOPSLA!

Apprenticeship Pedagogical Patterns — Sunday, 8:30-17:00 — *Royal Palm Salon 6*

http://softdevapprentice.org/ApprenticeshipPatterns.html

Pam Rostal, *New Mexico Highlands University, Synergenesis Consulting*
David West, *New Mexico Highlands University, University of New Mexico*
Joseph Bergin, *Pace University*
Jutta Eckstein, *IT communication*
Mary Lynn Manns, *University of North Carolina Asheville*
Linda Rising, *Linda Rising*

Best Practices for Model Driven Software Development — Sunday, 8:30-17:00 — *Royal Palm Salon 5*

http://www.softmetaware.com/oopsla2005/mdsd-workshop.html

Jorn Bettin, *SoftMetaWare*
Ghica van Emde Boas, *Independent Consultant*
Jean Bézivin, *University of Nantes*
Markus Voelter, *Independent Consultant*
William Cook, *University of Texas*

Beyond the Project Myth - Agile Development and Product Environments — Sunday, 8:30-17:00 — *Brittany*

http://www.coldewey.com/publikationen/conferences/oopsla2005/wkshpProduct.html

Jens Coldewey, *Coldewey Consulting*
Martin Lippert, *it-agile GmbH*
Klaus Marquardt, *Draeger Medical*

Building Software for Pervasive Computing — Sunday, 8:30-17:00 — *Royal Palm Salon 4*

http://www.ics.uci.edu/~lopes/bspc05/

Cristina Lopes, *University of California, Irvine*
Steffen Schaefer, *IBM Global Services*
Siobhan Clarke, *Trinity College, Dublin*
Tzilla Elrad, *Illinois Institute of Technology*
Jens Jahnke, *University of Victoria*

Early Aspects — Sunday, 8:30-17:00 — *Royal Palm Salon 3*

www.early-aspects.net

Elisa Baniassad, *Chinese University of Hong Kong*
Joao Araujo, *New University of Lisbon*
Paul Clements, *Carnegie Mellon University's Software Engineering Institute*
Paulo Merson, *Carnegie Mellon University's Software Engineering Institute*

Eclipse Technology eXchange (ETX) 2005—Day 1 — Sunday, 8:30-17:00 — *Golden West Room*

http://www.cs.uvic.ca/~mstorey/etx2005

Margaret-Anne Storey, *University of Victoria*
Michael Burke, *IBM Research*
Li-Te Cheng, *IBM Research*
Andre van der Hoek, *University of California, Irvine*

Extravagaria III: Hunting Creativity — Sunday, 8:30-17:00 — *Royal Palm Salon 2*

http://dreamsongs.com/Feyerabend/Extravagaria.html

Richard Gabriel, *Sun Microsystems, Inc.*
John Gribble, *Tokyo University of Fine Art and Music*
Brenda Laurel, *Art Center College of Design; Sun Microsystems*

Fostering Software Reliability in an Increasingly Hostile World

Sunday, 8:30-17:00 — ***Fairfield***

http://mysite.verizon.net/dennis.mancl/oopsla05

Dennis Mancl, *Lucent Technologies*
Steven Fraser, *Qualcomm*
Amir Zeid, *The British University in Egypt*
Greg Utas, *Pentennea LLC*

Fourth "Killer Examples" for Design Patterns and Objects First Workshop

Sunday, 8:30-17:00 — ***Clarendon***

http://www.cse.buffalo.edu/faculty/alphonce/KillerExamples/OOPSLA2005/

Carl Alphonce, *University at Buffalo, State University of New York*
Stephen Wong, *Rice University*
Michael Caspersen, *University of Aarhus*
Adrienne Decker, *University at Buffalo*

Library-centric software design

Sunday, 8:30-17:00 — ***Dover***

http://lcsd05.cs.tamu.edu

David Musser, *Rensselaer Polytechnic Institute*
Andrew Lumsdaine, *Indiana University*
Jaakko Jarvi, *Texas A&M*
Sibylle Schupp, *Chalmers University of Technology*
Todd Veldhuizen, *Chalmers University of Technology*

Third Workshop on Method Engineering for Object-Oriented and Component-Based Development

Sunday, 8:30-17:00 — ***Royal Palm Salon 1***

http://www.open.org.au/Conferences/oopsla2005/MEW.html

Magdy Serour, *University of Technology, Sydney*
Donald Firesmith, *Software Engineering Institute*
Pavel Hruby, *Microsoft Business Solutions*
Brian Henderson-Sellers, *University of Technology, Sydney*
Dan Rawsthorne, *NetObjectives*
Bernhard Rumpe, *Braunschweig University of Technology*
Cesar Gonzalez-Perez, *University of Technology, Sydney*
Hadar Ziv, *University of California, Irvine*

Synchronization and concurrency in object-oriented languages (SCOOL)

Sunday, 8:30-17:00 — ***California Room***

http://research.microsoft.com/~tharris/scool

Tim Harris, *Microsoft Research*
Doug Lea, *State University of New York*
David F. Bacon, *IBM Research*
Keir Fraser, *University of Cambridge*
Maurice Herlihy, *Brown University*
Michael Hicks, *University of Maryland*
Tony Hosking, *Purdue University*
Gary Lindstrom, *University of Utah*
Victor Luchangco, *Sun Microsystems Labs*
John Potter, *University of New South Wales*
Ravi Rajwar, *Intel*
Michael L Scott, *University of Rochester*

Croquet: A Platform for Collaboration

Monday, 8:30-17:00 — ***Eaton***

http://www.croquetproject.org

Alan Kay, *Hewlett-Packard and President, Viewpoints Research Institute, Inc.*
Julian Lombardi, *University of Wisconsin*
Mark McCahill, *University of Minnesota*
Rick McGeer, *Hewlett-Packard*
Andreas Raab, *Hewlett-Packard*
David P. Reed, *Hewlett-Packard and MIT*
David A. Smith, *Croquet Project*

Eclipse Technology eXchange (ETX) 2005—Day 2

Monday, 8:30-17:00 — ***Golden West Room***

http://www.cs.uvic.ca/~mstorey/etx2005

Margaret-Anne Storey, *University of Victoria*
Michael Burke, *IBM Research*
Li-Te Cheng, *IBM Research*
Andre van der Hoek, *University of California, Irvine*

The 5th OOPSLA Workshop on Domain-Specific Modeling

Monday, 8:30-17:00 ***Royal Palm Salon 1***

http://www.dsmforum.org/events/DSM05

Juha-Pekka Tolvanen, *MetaCase*
Jonathan Sprinkle, *University of California, Berkeley*
Matti Rossi, *Helsinki School of Economics*

International Workshop on Software Factories

Monday, 8:30-17:00 ***Royal Palm Salon 6***

http://www.softwarefactories.com/workshops/OOPSLA%202005/

Jack Greenfield, *Microsoft Corporation*
Steve Cook, *Microsoft Corporation*
Krzysztof Czarnecki, *University of Waterloo*
Jeff Gray, *University of Alabama at Birmingham*
Michael Stal, *Siemens AG*
Gabor Karsai, *Vanderbilt University*
Markus Voelter, *Independent Consultant*
Don Batory, *University of Texas at Austin*
Brian Henderson-Sellers, *University of Technology at Sydney*
Cesar Gonzalez-Perez, *University of Technology at Sydney*

Fourth International Workshop on Agent-Based Methodologies

Monday, 8:30-17:00 ***Royal Palm Salon 2***

http://www.open.org.au/Conferences/oopsla2005/AOW.html

Cesar Gonzalez-Perez, *University of Technology, Sydney*
Paolo Bresciani, *ITC-Irst*
Monique Calisti, *Whitestein Technologies*
John Debenham, *University of Technology, Sydney*
Paolo Giorgini, *University of Trento*
Ian Gorton, *National ICT Australia*
Brian Henderson-Sellers, *University of Technology, Sydney*
Graham Low, *University of New South Wales*

Java Technologies for Real-Time and Embedded Systems

Monday, 8:30-17:00 ***Royal Palm Salon 4***

http://www.cs.wustl.edu/~corsaro/JTRES05

Greg Bollella, *Sun Microsystems*
Angelo Corsaro, *Alenia Marconi Systems*
Peter Dibble, *TimeSys*
Doug Lea, *State University of New York at Oswego*
Corrado Santoro, *University of Catania*
Jan Vitek, *Purdue University*
Andy Wellings, *University of York*

Multiparadigm Programming in Object-Oriented Languages

Monday, 8:30-17:00 ***Fairfield***

http://www.c3.lanl.gov/mpool05/

Kei Davis, *Los Alamos National Laboratory, NM*
Joerg Striegnitz, *Xcc Software AG, Karlsruhe*
Gavin Bierman, *Microsoft Research, Cambridge*
Timothy Budd, *Oregon State University*
Jaakko Jarvi, *Texas A&M University*
Herbert Kuchen, *University of Muenster*
Peter Van Roy, *Catholic University of Louvain*

Third Int'l Workshop on SOA and Web Services Best-practices

Monday, 8:30-17:00 ***Royal Palm Salon 3***

http://www.arsanjani.com/oopsla2005/soa

Ali Arsanjani, *IBM Corporation*
Kerrie Holley, *IBM Corporation*
Amir Zeid, *The British University in Egypt*

Scrapheap Challenge—A Workshop in Post-Modern Programming

Monday, 8:30-17:00 ***Clarendon***

http://www.xpdeveloper.com/xpdwiki/Wiki.jsp?page=ScrapheapChallenge

Ivan Moore, *ThoughtWorks Ltd*
Nat Pryce, *ThoughtWorks Ltd*

MVCDC 2: Managing Variabilities consistently in Design and Code

Thursday, 8:30-17:00 ***Esquire***

http://www.kircher-schwanninger.de/workshops/MVCDC/

Christa Schwanninger, *Siemens AG*
Danilo Beuche, *pure-systems GmbH*
Markus Voelter, *ingenieurbüro für softwaretechnologie*
Krzysztof Czarnecki, *University of Waterloo*
Mira Mezini, *Darmstadt University of Technology*
Rainer Burgstaller, *Darmstadt University of Technology*

ACM / SIGPLAN / SIGSOFT

ACM, the Association for Computing Machinery, founded in 1947, is the world's first educational and scientific computing society. Today, its members – over 75,000 computing professionals and students worldwide – and the public turn to ACM for authoritative publications, pioneering conferences, and visionary leadership for the new millennium. ACM offers its members a vast array if IT information resources, including the ACM Portal, which includes over 750,000 pages of text, the Online Guide to Computing Literature (with 350,000 bibliographic citations extending far beyond ACM's own literature), the Online computing Review Service, customized personal services and more. For a complete listing of ACM's Professional and Student memberships benefits and options, visit http://www.acm.org/joinacm.

ACM SIGPLAN explores programming languages concepts and tools, focusing on design, implementation, and efficient use. Its members are programming language users, developers, implementers, theoreticians, researchers and educators. The monthly newsletter "ACM SIGPLAN Notices" publishes several conference proceedings issues, regular columns and technical correspondence. This SIG offers an additional newsletter (FORTRAN forum), on a subscription only basis. SIGPLAN sponsors four major annual conferences: the OOPSLA conference on object-oriented technology, the Conference on Programming Language Design and Implementation (PLDI), the major professional conference in the field, the Symposium on Principles of Programming Languages (POPL), and the International Conference on Functional Programming (POPL), held periodically in Europe.

Contact ACM to Join

Voice:

1-800-342-6626: US & Canada
1-212-626-0500 NY & Global

Fax:

1-212-944-1318

Email:

acmhelp@acm.org

Web:

http://www.acm.org/
http://www.acm.org/join
http://www.acm.org/sigplan
http://www.acm.org/sigsoft

OOSPLA 2005 Committees

Conference Committee

Conference Chair
Ralph Johnson, *University of Illinois*
Program Chair
Richard P. Gabriel, *Sun Labs*
Communications Chair
Gail E. Harris, *Instantiated Software*
Corporate Support Chair
Shail Arora, *Gradepoint, Inc.*
Demonstrations and Exhibits Chair
Ken Bauer, *ITESM Campus Guadalajara*
DesignFest Chair
Daan Hoogland, *Luminis B.V.*
Doctoral Symposium Chair
Fred Grossman, *Pace University*
Educators' Symposium Chair
Eugene Wallingford, *University of Northern Iowa*
Essays Chair
Brian Marick, *Exampler*
Lightning Talks co-chairs
Bjorn Freeman-Benson, *Predictable Software*
Bruce Horn, *Ingenuity Software*
Onward! co-chairs
Elisa Baniassad, *The Chinese University HK*
James Noble, *Victoria University of Wellington, NZ*
Panels Chair
Steve Berczuk, *Iron Mountain*
Posters Chair
Dirk Siebert
Practitioner Reports Chair
Robert Biddle, *Carleton University*
ACM Student Research Competition Chair
Dirk Siebert
Student Volunteers Chair
Maria Elena (Helen) Chávez Echeagaray, *Tecnológico de Monterrey-Campus Guadalajara*
Student Volunteers Captain
Yolanda Cham Yuen, *Tecnológico de Monterrey-Campus Guadalajara*
Steering Committee Chair
Mahmdouh Ibrahim, *IBM*
Technology Chair
Max Rahder, *Rahder Consulting*
Treasurer
Vicki Hanson, *IBM Research*
Tutorials Chair
Joe Bergin, *Pace University*
Workshops Chair
Bill Opdyke, *North Central College*
OOPSLA 2006 Chair
Peri Tarr, *IBM*
Liaison with ACM
Erin Dolan, *ACM*
Operations
Erin Peterson, *Meeting Strategies Worldwide*
Registration
Carole Mann, *Registration Systems*
CyberChair Operation and Support
Richard van de Stadt, *Borbala*
Webmaster
Ken Bauer, *ITESM Campus Guadalajara*

Program Committee

Alan O'Callaghan, *De Montfort University*
Andrew P. Black, *Portland State University*
Christa Schwanninger, *Siemens AG*
Christoph Kirsch, *Universität Salzburg*
Cristina Videira Lopes, *University of California, Irvine*
Daniel May, *Independent*
David Bacon, *IBM Research*
David Clarke, *Centrum voor Wiskunde en Informatica*
David Thomas, *Bedarra Corporation*
David West, *New Mexico Highlands University*
Dirk Riehle, *Independent Consultant*
Douglas C. Schmidt, *Vanderbilt University*
Greg Morrisett, *Harvard University*
Julie McCann, *Imperial College*
Karl Lieberherr, *Northeastern University*
Kathleen Fisher, *AT&T Labs*
Kent Beck, *Three Rivers Institute*
Linda G. DeMichiel, *Sun Microsystems*
Martin Rinard, *MIT*
Mattias Felleisen, *Northeastern University*
Michael Kölling, *University of Kent*
Peter Sommerlad, *Hochschule Für Technik*
Richard P. Gabriel, *Sun Labs*
Robert Biddle, *Carleton University*
Robert Walker, *University of Calgary*
William Cook, *University of Texas*
William Pugh, *University of Maryland*

OOSPLA 2005 Committees

ACM Student Research Competition Committee
Dirk Siebert (Chair)
Daan Hoogland, *Luminis B.V., The Netherlands*
Lucas Layman, *North Carolina State University*
Trevor Parsons, *University College Dublin, Ireland*
Tao Xie, *North Carolina State University*

DesignFest Committee
Daan Hoogland (Chair), *Luminis B.V.*
Andrew Eland, *Google*
Ghica van Emde Boas, *Bronstee.com Software & Services*
Jim Heliotis, *Rochester Institute of Technology*
Matthew S. Davis, *The Pennsylvania State University*
Peter Pascale, *Pearson VUE*

Doctoral Symposium Committee
Fred Grossman (Chair), *Pace University*
Craig Chambers, *University of Washington*
Ron Frank, *Pace University*
Robert Kessler, *University of Utah*
Ole Madsen, *Alexandra Instituttet A/S, Aarhus University*

Educators' Symposium Committee
Owen Astrachan, *Duke University, USA*
Robert Biddle, *Carleton University, Canada*
Robert Duvall, *Duke University, USA*
Jutta Eckstein, *Independent Consultant, Germany*
Mary Lynn Manns, *UNC Asheville, USA*
Rick Mercer, *University of Arizona, USA*
Chris Nevison, *Colgate University, USA*
Eugene Wallingford, *University of Northern Iowa*

Essays Committee
Andrew Hunt, *Pragmatic Programmers, LLC*
Andrew Pickering, *University of Illinois*
Brian Marick, *Testing Foundations*
Carey Schwaber, *Forrester Research, Inc.*
John Holbo, *National University of Singapore*
Lucy Suchman, *Lancaster University*
Mark E. Johnson, *University of Central Florida*
Peter Turchi, *Warren Wilson College*
Rebecca J. Parsons, *ThoughtWorks, Inc.*
Robert Austin, *Harvard Business School*
Rodney Brooks, *MIT*

Lightning Talks Committee
Bjorn Freeman-Benson, *Eclipse Foundation*
Bruce Horn, *Ingenuity Software*

Onward! Committee
David Ungar, *Sun Labs*
Elisa Baniassad, *The Chinese University of Hong Kong*
James Noble, *Victoria University of Wellington*
Jonathan Maletic, *Kent State University*
Mary Poppendieck, *Poppendieck, LLC*
Ole Lehrmann Madsen, *Aarhus Universitet*
Yvonne Coady, *University of Victoria*

Posters Committee
Dirk Siebert (Chair)
Jill Aden, *EDS*
Uri Dekel, *Carnegie Mellon University*
Ada Diaconescu, *Dublin City University, Ireland*
Hector Gerardo Perez Gonzalez, *Universidad Autónoma de San Luis Potosí, Mexico*
Laura Hill, *Sun Microsystems Laboratories*
Daan Hoogland, *Luminis B.V., The Netherlands*
Martin Lippert, *it-agile GmbH, Germany*
Mary Lynn Manns, *University of North Carolina at Asheville*

Practitioner Reports Committee
Robert Biddle (Chair), *Carleton University, Canada*
Neville Churcher, *University of Canterbury, New Zealand*
Sally Jo Cunningham, *University of Waikato, New Zealand*
Rachel Davies, *Agile Experience, UK*
Alain Désilets, *National Research Council, Canada*
Tom Poppendieck, *Poppendieck LLC., USA*
Max Rahder, *Rahder Consulting, USA*
J. B. Rainsberger, *Diaspar Software Services, Canada*

Steering Committee
Mamdouh Ibrahim (Chair), *IBM*
Linda Northrop, *Software Engineering Institute (Past Chair)*
Ron Crocker, *Motorola*
John Vlissides, *IBM T. J. Watson Research Center*
Ralph Johnson, *University of Illinois*
Richard Gabriel, *Sun Labs*
Peri Tarr, *IBM T. J. Watson Research Center*
Jack Davidson, *University of Virginia*
Kathleen Fisher, *AT&T Labs*
Erin Dolan, *ACM*

Tutorials Committee
Joe Bergin (Chair), *Pace University*
Steve Metsker, *Reader, Writer, & Teacher*
Alan Knight, *Cincom Systemst*
Angelika Langer, *Trainer/Consultant*
Rebecca Wirfs-Brock, *Wirfs-Brock Associates*

Workshops Committee
Bill Opdyke (Chair), *North Central College*
Steven Fraser, *Qualcomm*
Doug Lea, *State University of New York at Oswego*
Dennis Mancl, *Lucent Technologies*
Doug Schmidt, *Vanderbilt University*
Helen Sharp, *The Open University, UK*

Notes

Notes

About San Diego

About San Diego

There are many options for places to have a meal:

5 onsite restaurants

Sunshine Deli (open daily 7am-7pm)
Terrace Cafe (open daily at 6am)
Charlie's Sports Bar (open for dinner/evenings)
Trellises Garden Grille (open for breakfast, lunch & dinner)
Kelly's Steakhouse (open for dinner/evenings)

Fashion Valley Food Court

The Cheesecake Factory
PF Chang's China Bistro
Crocodile Cafe
California Pizza Kitchen
Pizzeria Uno and
twelve bistro cafes.

Visit Old Town

Numerous restaurants

Old Town is 2 trolley stations from the conference center and costs just $1.75 for a one way fare. You can catch the trolley at the Fashion Valley Transit Center, across the bridge at the North end of the property. You may purchase trolley tickets from the vendomats there. Some machines require exact change; some accept $1.00 or $5.00 bills. Susan B. Anthony coins are also accepted. See San Diego Trolley Information for more information.

You can ask the Town & Country concierge for directions to the Fashion Valley Transit Center and to Old Town.

Airport

San Diego International Airport
(approx. 10-15 min. drive to T&C hotel)
www.san.org

Los Angeles International Airport
(approx. 110 miles (177km) to San Diego)
www.lawa.org/lax/welcomeLAX.cfm

Cabs/Taxis

Airport Yellow Cab of San Diego, 619.234.6161 or
www.driveu.com/AirportYellow.asp

Orange Cab, 619.291.3333 or
www.orangecabsandiego.com

Shuttles (also Towncar & Limo services)

Xpress Shuttle (also has towncar service)
800.900.RIDE (7433) or www.xpressshuttle.com/
san_diego.htm

Cloud 9 Shuttle (also has towncar or limo service)
800.9.SHUTTLE (748.8853) or www.cloud9shuttle.com

Rental Cars

Avis Rent-a-Car. 800.852.4617 or www.avis.com

Budget Rent-a-Car, 800.527.0700 or www.budget.com

Hertz Rent-a-Car, 800.654.3131 or www.hertz.com

National Car Rental, 800.CAR.RENT (227.7368) or www.nationalcar.com

Thrifty Rent-a-Car, 800.847.4389 or www.thrifty.com

San Diego Trolley Information

www.sdcommute.com/Rider_Information/trolley/index.asp

Things to do in San Diego

Riverwalk Golf Course
(located next to the Town & Country)
1150 Fashion Valley Road
San Diego, California 92108
http://san-diego-golf.com/riverwalk.html
Located in the heart of Mission Valley, Riverwalk opened to the public on March 16th, 1998. It features a 27-hole championship golf course. The facilities include a lighted double-sided driving range with target greens (open every day until 10pm); a new clubhouse with a fully equipped golf shop and an elegant restaurant, and tournament banquet area. Premium rental equipment is also available.

Fashion Valley Shopping Mall
(located across the street from the Town & Country)
7007 Friars Road
San Diego, CA 92108
http://www.simon.com/mall/default.aspx?ID=765
Beautiful outdoor shopping center featuring 200 stores & restaurants, including six department stores, over 40 stores that are exclusive to the area, a delightful food court, and an 18 screen AMC Theatre.

Take the Trolley to Old Town
Old Town San Diego is considered the "birthplace" of California. San Diego is the site of the first permanent Spanish establishment in California (1769). Old Town boasts the finest in theatre, artists, galleries, shops & historical sites - all within easy walking distance. Visit the original and reconstructed buildings and furnishings that illustrate the ambiance of the 1800s San Diego.

Take the Trolley to Downtown San Diego/Seaport Village
(6 Trolley Stations from the Fashion Valley Transit Ctr.)

More Things to Do in the San Diego Area:

San Diego Zoo: www.sandiegozoo.com

SeaWorld Adventure Park: www.seaworld.com

Legoland: www.lego.com/legoland

Balboa Park: www.balboapark.org

Conference Chair: Peri Tarr, *IBM Research*

On behalf of the OOPSLA 2006 Conference Committee, I invite you to attend, contribute to, and participate in the 21st OOPSLA, to be held in Portland, Oregon, USA, October 22-26, 2006, at the Oregon Convention Center. OOPSLA is widely recognized as the premier gathering of practitioners, researchers, educators, and students sharing their thoughts and experiences with object technologies.

Now more than ever, OOPSLA is also a haven where contributors and attendees have a chance to make sense of an alphabet soup of acronyms, terms, and technologies—Java, .NET, services, XML, BPEL, WSDL, MDD...the list goes on and on. You know what happens when you try to synthesize and integrate today's technologies—some object-oriented, some not—to accomplish real-world goals: challenges, frustrations, and delays arise immediately. We must find ways to slice through the technology fog, bringing coherence to software development. The result could be "the next big thing" in computer science. At OOPSLA'06 the spotlight will be on this theme of "coherence."

OOPSLA offers many ways to participate, including technical papers, practitioner reports, expert panels, posters, workshops, demonstrations, birds-of-a-feather sessions, as well as plenty of social opportunities for networking, mingling, munching, and just having fun.

Becoming an active participant in OOPSLA is easier than you might think – identify the events that match your interests and work, review the submission requirements from the OOPSLA 2006 web site, and submit your contribution using the OOPSLA on-line submission system. If you have ideas for further OOPSLA activities, such as one-day symposia or student activities, please email a proposal to the Conference Chair for consideration.

We strongly encourage you to contribute to OOPSLA 2006 and to take part in shaping the future of object technology. See you in Portland!

Student Volunteer Program

The Student Volunteer program is an opportunity for students from around the world to associate with the top people in object-oriented technology, research and software development. In return for about ten hours of their time, student volunteers receive complimentary registration, free admission to tutorials (space permitting), and other benefits. Contact the Student Volunteer Chair at sv@oopsla.org for more information.

OOPSLA 2006 Important Dates

March 18, 2006

Submission deadline for Technical Papers, Onward! Papers, Essays, Practitioner Reports, Educators' Symposium Submissions, Tutorials Proposals, Panels Proposals, Workshops Proposals, and DesignFest Proposals.

June 30, 2006

Submission deadline for Lightning Talks, Posters, Demonstrations, Doctorial Symposium, and Student Volunteers.

OOPSLA 2006 Contact Information

Conference Chair: Peri Tarr, IBM Research, chair@oopsla.org
Program Chair: William Cook, University of Texas at Austin, papers@oopsla.org

For information, please contact:

ACM Member Services Department
1-800-342-6626 (US & Canada)
1-212-626-0500 (Global)
E-mail: info@oopsla.org

Facility Map