ooPSLA is no ordinary conference. It's not even the same conference it was ten years ago—or five. ooPSLA started as a conference devoted to object-oriented programming, but it has always been more. It's the place where people devoted to the art and craft of software development gather to share ideas, to push the boundaries, and to learn and create. It's about programming, but not just the cold details of a language. It's about language, but not just about syntax and semantics and type systems. It's about systems and applications, collections of programs that work in the world, but not just about finished products. It's about all of these things, about the people who make software and the people who use it. It's about how to build programs, well and better. It's about making systems and applications that make the world a better place for people.

ooPSLA's still the #1 conference for learning and extending OO and its related technologies. Yet ooPSLA evolves—as the software industry creates new ideas and encounters new challenges, as researchers discover new ways to talk about programs and programming languages. When you come to ooPSLA, you won't be “back when,” but “what’s now” and “what’s next?” You will leave the conference very excited and emboldened for today’s challenges and ready for tomorrow’s advances.

We are proud to offer you ooPSLA and look forward to sharing the experience with you in Montréal this fall.

— Richard P. Gabriel

WHAT’S GOING ON?

John McCarthy. Patti Maes. Gregor Kiczales. Fred Brooks. We all recognize these as the names of researchers who have made profound contributions to computing and software. They are also the headline speakers at this year’s ooPSLA, where you’ll hear what they are thinking about in 2007. ooPSLA invited talks have a well-deserved reputation for energizing software developers with ideas that stretch our collective mind.

Jim Purbrick and Mark Lentczner. You may not recognize these names, but you might know their alter egos: Babbage Linden and Zero Linden, of Linden Labs. They are the creators of Second Life, a 3D virtual world that is changing how we think about the real world, how we live in multiple worlds, and how we do business in the 21st century. Purbrick and Lentczner are the Onward! keynote speakers.

Peter Turchi. He is the director of the MFA Program for Writers at Warren Wilson College. You certainly don’t recognize his name. Turchi is the award-winning author of Maps of the Imagination, an exploration of the relationships between creative writing, making maps, and creativity. He is our keynote speaker, and perhaps the most mind-bending of all our speakers.

— Robert Biddle

REGISTRATION

Register online at http://www.oopsla.org

September 13 – Last day for reduced rate
October 11 – Last day for advance registration
ooPSLA newcomers and veterans alike face a challenge: with so much going on, and so much to learn, how can you make the most of your time at the conference? There are no easy answers. But, knowing what each track offers is a good first step.

Workshops are like mini-conferences on a particular topic. Usually, you need to contact the workshop organizers in advance to participate. Tutorials are half-day courses on topics of industrial and academic importance; think of them as continuing education for programmers. Tutorials are the only ooPSLA events that require an extra fee. Other tracks offer just what their name suggests. Demos allow you to see software tools in action, as presented by their creators, while practitioner reports share the experiences gained in industry when companies implement ideas that change how they do their job. The educators symposium is a full day focused on teaching computer science and software development. And Onward! is the conference within a conference where presenters discuss big thoughts that challenge the status quo of how we think about and make software. You can learn more about these tracks and other ooPSLA tracks at the conference web site.

The key to navigating ooPSLA is realizing that throughout the conference you will have a chance to participate in all of these tracks. The symposia happen early in the week, but in any given time slot you will have the chance to hear research talks, attend a demo, go to a tutorial, hear an Onward! talk or a practitioner report, or hear an essayist or invited speaker present their take on our discipline. As the conference nears, visit the web site for pointers to events organized by topic across all of the tracks. And don’t forget one of the major benefits of ooPSLA, one that doesn’t appear in the printed program: the opportunity to interact with professionals and academics who are working at the heart of programming. Be sure to save some energy for coffee breaks, ad hoc conversations in the halls, and more relaxed time over lunch or dinner talking shop and learning from the wide range of folks who make ooPSLA a must-attend conference on their calendar each year.

– Eugene Wallingford

ooPSLA newcomers and veterans alike face a challenge: with so much going on, and so much to learn, how can you make the most of your time at the conference? There are no easy answers. But, knowing what each track offers is a good first step.

Workshops are like mini-conferences on a particular topic. Usually, you need to contact the workshop organizers in advance to participate. Tutorials are half-day courses on topics of industrial and academic importance; think of them as continuing education for programmers. Tutorials are the only ooPSLA events that require an extra fee. Other tracks offer just what their name suggests. Demos allow you to see software tools in action, as presented by their creators, while practitioner reports share the experiences gained in industry when companies implement ideas that change how they do their job. The educators symposium is a full day focused on teaching computer science and software development. And Onward! is the conference within a conference where presenters discuss big thoughts that challenge the status quo of how we think about and make software. You can learn more about these tracks and other ooPSLA tracks at the conference web site.

The key to navigating ooPSLA is realizing that throughout the conference you will have a chance to participate in all of these tracks. The symposia happen early in the week, but in any given time slot you will have the chance to hear research talks, attend a demo, go to a tutorial, hear an Onward! talk or a practitioner report, or hear an essayist or invited speaker present their take on our discipline. As the conference nears, visit the web site for pointers to events organized by topic across all of the tracks. And don’t forget one of the major benefits of ooPSLA, one that doesn’t appear in the printed program: the opportunity to interact with professionals and academics who are working at the heart of programming. Be sure to save some energy for coffee breaks, ad hoc conversations in the halls, and more relaxed time over lunch or dinner talking shop and learning from the wide range of folks who make ooPSLA a must-attend conference on their calendar each year.

– Eugene Wallingford

Imagine yourself in 1986. You’ve done battle with Brain, the first PC virus. Halley’s comet has visited our solar system for the second time this century. Vancouver, Canada, is host to the World Exposition. And you are attending ooPSLA, the first conference devoted to object oriented programming. Fast forward to 2007. Even your grandmother knows how to run virus protection software. Comet McNaught, the brightest comet to appear in over 40 years, was visible over the Southern Hemisphere.

Throughout this period, ooPSLA has been a huge success. Early works on dynamic compilation, design patterns, and aspect oriented programming—first published here—have evolved and grown into conference topics in their own right. Java, heavily influenced by ooPSLA, is one of the most popular programming languages today. Refactoring, eXtreme Programming, Unified Modeling Language, and many other significant technologies instilled in popular practice through ooPSLA are success stories of which we can all be proud.

Come join us in Montréal to see what new and exciting developments are on the horizon for the next twenty years.

– Gail E. Harris
TUTORIALS

1 Software Evolution: Analysis and Visualization
   Harald Gall, University of Zurich

2 The Web: Distributed Objects Realized!
   Stuart Charlton, BEA Systems

3 Revolutionizing Software Quality through Static Analysis Tools
   Jonathan Aldrich, Carnegie Mellon University

4 Planning and Executing Agile Projects in Non-Agile Environments
   Peter Schub, Independent Consultant

5 High Quality Software Architecture
   Michael Stal, Siemens AG Corporate Research and Technology

6 Principles of Aspect-Oriented Design in Java and AspectJ
   Dean Wampler, Object Mentor, Inc.

7 Unit Test Patterns and Smells: Improving Test Code and Testability Through Refactoring
   Gerard Meszaros, ClearStream Consulting

8 From Use to User Interface: Collaboratively designing, prototyping, and testing user interface to help your users succeed
   Jeff Patton, ThoughtWorks

9 Pick Up Your Pen!
   Steve Metsker, Dominion Digital

10 Object-Oriented Models of Requirements and High-level Software Design
    Hermann Kaindl, Vienna University of Technology, IGT

11 Best Practices for Model-Driven Development
    Markus Voelter, Independent Consultant

12 Software Architecture Refactoring
    Michael Stal, Siemens AG Corporate Research and Technology

13 LINQ From the Source
    Erik Meijer, Microsoft Corp

14 Security Patterns and Secure Software Architecture
    Munawar Hafiz, University of Illinois at Urbana-Champaign

15 Java API Design
    Boris Bukowski, IBM

16 Test-Driven Development—Hands-on!
    Niclas Nilsson, factor/10

17 Transformation and Analysis of the Java Bytecode with ASM Framework
    Eugene Kuleshov, independent

18 Software Architecture Documentation in the Real World
    Markus Voelter, Independent Consultant

19 Introduction to Concurrent Programming in Java
   David Holmes, Sun Microsystems, Inc.
   Joe Boubker, Mobile App Consulting

20 The Scala Experience—programming with functional objects.
   Martin Odersky, EPFL
   Ted Neward, self
   Gilles Dubachet, EPFL

21 Personas, Profiles, Actors, & Roles: Modeling users to target successful product design
   Jeff Patton, ThoughtWorks

22 Building Embedded and Stand-alone Domain Specific Languages: Principles and Practice
   Eric Van Wyk, University of Minnesota

23 Using Java ME to Program Your Mobile Phone
   Jonathan Knudsen, Sun Microsystems, Inc.

24 The Art of Telling Your Design Story
   Rebecca Wirfs-Brock, Wirfs-Brock Associates

25 Integrating Architecture-Centric Methods into Object-Oriented Analysis and Design
   Ragbyvinder Sanguan, Pennsylvania State University

26 Why Users Say, “Start with the Screen!”: Effective Test-Driven Development, Presentation Layer-First
   Bobby Norton, ThoughtWorks
   Chris Stevenson, ThoughtWorks

27 SOA in Reality
   Nicolai Josuttis, self employed

28 Java Concurrency Utilities in Practice
   David Holmes, Sun Microsystems, Inc.
   Joe Boubker, Mobile App Consulting

29 Using FindBugs in Anger
   David Hovemeyer, York College of Pennsylvania
   William Pugh, Univ. of Maryland

30 Incremental Releases Users & Stakeholders Will Love
   Jeff Patton, ThoughtWorks

31 Green Bar for C++—Unit Testing and Refactoring C++
   Peter Sommerlad, IBM Institute for Software at HSR Rapperswil

32 Writing Adaptable Software: Mechanisms for Implementing Variabilities in Code and Models
   Markus Voelter, Independent Consultant

33 Software Best-Practices: Agile Deconstructed
   Steven Fraser, Cisco Systems

34 The WS-* Jungle
   Jorg Bartholdt, Siemens
   Michael Stal, Siemens

35 Scripting Your (and Your Company’s) Second Life
   Cristina Lopes, UC Irvine
   William Cook, UT Austin

36 Agile in the Face of Global Software Development
   Jutta Eckstein, IT communication

37 Use-Case Patterns and Blueprints
   Gunnar Overgaard, SEB AB
   Karin Palmqvist, Generic Integration AB

38 Embedded Objects with C++: Idioms, Patterns, and Architecture for Constrained Systems
   Detlef Vollmann, vollmann engineering gmbh

39 Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing
   Doug Schmidt, Vanderbilt University

40 Software Security: Building Security In
   Gary McGraw, Cigital

41 Behaviour-Driven Development using JBehave
   Elizabeth Keogh, Thoughtworks Ltd.
   Dan North, Thoughtworks Ltd.

42 Fault Tolerant Software with Patterns
   Robert Hamer, Alcatel-Lucent

43 Testing: It’s Part of Your Job—Make it Effective!
   Neil Harrison, Utah Valley State College

44 Skills for the Agile Designer
   Rebecca Wirfs-Brock, Wirfs-Brock Associates

45 Domain-Driven Design: Putting the Model to Work
   Eric Evans, Domain Language, Inc.

46 Real-time Programming on the Java Platform
   David Holmes, Sun Microsystems, Inc.
   Tony Printezis, Sun Microsystems, Inc.

47 Totally Awesome Computing: Python as a General-purpose Object-oriented Programming Language
   Chuck Allison, Utah Valley State College

48 Introduction to Software Product Line Development Methods
   Charles Kreuger, BigLever Software

49 Creating Plugins and Applications for the Eclipse Platform
   Alex Romanow, Radview and Tel-Aviv University
   Amir Kirsh, Converse and Academic College of Tel-Aviv Yaffo

50 Domain-Driven Design: Strategic Design for Larger Systems
   Eric Evans, Domain Language, Inc.

51 Introduction to Roby
   Glenn Vanderburg, Independent Consultant

52 Building Service-Oriented Architectures with Web Services
   Olaf Zimmermann, IBM Research

53 UML in Action
   Mohamed E. Fayad, San Jose State University & vrlSoft, Inc.
“I think ooPsla is the best d•mned programming conference in the world: there is nothing else like it. ooPsla is where people learn what they need today—and learn what they will be doing tomorrow. This is as true for programmers in the trenches as it is for researchers and academics and educators.”

- Long-time ooPsla Attendee / Iconoclast

ooPsla was founded in 1986 by the earliest of early adopters: researchers and practitioners thrilled by the prospects of object-oriented programming. Now, as we convene for the twenty-second time, objects are utterly mainstream, the very foundation of a great deal of the world’s software. But the thrill of creation is addictive, and ooPsla has developed a strong tradition of being the spawning ground for new ideas, artifacts, and movements—things like software patterns, aspects, refactoring, reflection, Eclipse, UML, and the Agile methods. At ooPsla, languages like Smalltalk, CLOS, C++, Beta, Self, Eiffel, C#, and Java have sprouted and bloomed, contributing their underlying ideas and expressive power to the pool of computing concepts. ooPsla strives to mingle people on the vanguard of research, and practitioners in the trenches who are reflecting upon and trying to understand just about every facet of software and programming.

http://www.oopsla.org