



Association for  
Computing Machinery

*Advancing Computing as a Science & Profession*

## NEWS RELEASE

**CONTACT:** Jim Ormond  
212-626-0505  
[ormond@hq.acm.org](mailto:ormond@hq.acm.org)

### **ACM SPLASH 2016 Conference Features Latest Developments in Software Construction and Delivery**

*Annual Event Features Presentations of Leading-Edge Research in  
Programming Languages and Software Engineering*

**New York, NY, October 26, 2016** – Software is both ubiquitous in our daily lives and a key driver of innovation in computing and technology today. From prototype automated vehicles to advanced humanoid robots, and from browsing the Internet to downloading music, software development teams are critical to the success of these systems.

The latest innovations in software development will be showcased at the ACM SIGPLAN Conference on Systems, Programming, Languages and Applications: Software for Humanity (SPLASH), to be held this year from October 30 – November 4 in Amsterdam. SPLASH is the premier conference at the intersection of programming languages and software engineering.

“At SPLASH, we present original research papers that address all aspects of software construction and delivery, as well as programming languages and software engineering.” said Eelco Visser, Professor at Delft University of Technology and SPLASH General Chair. “SPLASH is an umbrella event that provides a multitude of perspectives on the field through some 30 tracks, including five conferences, two symposia, as well as a lively collection of keynotes, demonstrations, tutorials, workshops, and invited talks.”

“The selection of papers in this year’s conference showcases the breadth and depth of research in programming languages and software development,” added Yannis Smaragdakis of the University of Athens, OOPSLA track Program Chair. “The top topic descriptors for accepted papers are a good illustration of the conference’s flavor: language design, language implementation, concurrency, program analysis, tools, compilers, programming models and paradigms, and type systems and logics.”

#### **SPLASH 2016 Highlights Include:**

##### **Keynote Addresses (partial list)**

**Martin Odersky, École Polytechnique Fédérale de Lausanne, “From DOT to Dotty: Foundations and Types for Objects as Modules”**

Odersky will present DOT, a particularly simple calculus that can express systems following these principles. DOT has been developed as the foundation of the next version of Scala. Odersky will also

report on *dotty*, a new Scala compiler that implements the constructs of DOT in its core data structures and that synthesizes object-oriented and functional programming concepts in several interesting new patterns.

**Guy L. Steele Jr., Oracle Labs, “How to Tell a Compiler What We Think We Know”**

Steele has been repeatedly quoted (and tweeted) as having remarked more than once over the last decade, “If it’s worth telling yourself (or another programmer), it’s worth telling the compiler.” In this talk, he will try to explain in more detail what he meant by this. For example, he has noticed that programming languages provide lots of ways to annotate one thing, but not very many good ways to talk about relationships among multiple things (other than regard to one as a “server” to which an annotation is attached and the others as “clients”).

**Simon Peyton Jones, Microsoft Research, Cambridge, “The Dream of a Lifetime: Shaping How our Children Learn Computing”**

In 2014, the National Curriculum in England was re-launched. It now requires that every child learns computer science, including programming, from primary school onwards. In his talk, Peyton Jones will sketch the UK experience with computer science education. He will discuss the challenges educators in the UK face, and in particular, he will elaborate on the desperate need for help from researchers in programming languages and computing education.

**Invited Talks (*partial list*)**

**Robert Grimm, Goldman Sachs, “Adventures in Software Evolution”**

Successful software applications, even when diligently maintained, tend towards the complex and brittle with the passage of time. This perpetual challenge of software engineering is the direct result of two fundamental forces that reinforce each other. First, the rate of change in computing is faster than in almost any other human endeavor, resulting in constant pressure to support more features. Second, the primacy of not disrupting production means that code, once released, is exceedingly hard to modify, let alone decommission. In this talk, Grimm presents two complementary projects at Goldman Sachs that seek to meet this challenge head-on.

**Alan Blackwell, University of Cambridge, “PLATEAU Keynote: How to Design a Programming Language”**

Programming languages—how to tell a computer what to do—are the core technology of the digital revolution, just as the invention of the wheel was the core technology enabling the design of land transportation systems, and the invention of the roof was the core technology enabling the design of cities. This talk draws lessons from the design of cars and cities to propose principles and processes for programming language design, as well as research agendas that will support those principles and processes.

**Tudor Gîrba, feenk.com, “Software Environmentalism”**

We produce software systems at an ever increasing rate. On the one hand, this is great. On the other hand, our ability to get rid of older systems does not keep up with that pace. Gîrba argues that we cannot continue to disregard how we will deal with software systems at a later time. He introduces the concept of software environmentalism as a systematic discipline to pursue and achieve habitability. He argues that

by building our systems with recycling in mind, we will be better able to understand software systems in the future and be able to reuse and evolve them as needed.

### **SPLASH-E**

The SPLASH Education (E) initiative will be a one-day workshop designed to enumerate the foundational concepts of computational education. Participants will be invited to create a lesson plan to teach a specific computational concept to 8-year olds.

### **Programming Languages and Mentoring Workshop**

The purpose of **Programming Languages Mentoring Workshop (PLMW)** is to give promising students who consider pursuing a graduate degree in this field an overview of what research in this field looks like and how to get into and succeed in graduate school. The program of PLMW will include talks by prominent researchers of the field of programming languages and software engineering providing an insight in their research.

For a full conference program please visit [SPLASH 2016](#).

### **About ACM SIGPLAN**

SIGPLAN ([www.sigplan.org](http://www.sigplan.org)) is a Special Interest Group of ACM that focuses on Programming Languages. In particular, SIGPLAN explores the design, implementation, theory, and efficient use of programming languages and associated tools. Its members are programming language users, developers, implementers, theoreticians, researchers and educators.

### **About ACM**

ACM, the Association for Computing Machinery ([www.acm.org](http://www.acm.org)), is the world's largest educational and scientific computing society, uniting computing educators, researchers and professionals to inspire dialogue, share resources and address the field's challenges. ACM strengthens the computing profession's collective voice through strong leadership, promotion of the highest standards, and recognition of technical excellence. ACM supports the professional growth of its members by providing opportunities for life-long learning, career development, and professional networking.

###