EDUCATORS' SYMPOSIUM

Celebrating the 10th anniversary of the Educators' Symposium—the unique forum for trainers and educators.

Chair: Jutta Eckstein, *Objects in Action*

The Educators' Symposium is for industry and academic professionals who have a vested interest in object technology training and education. This one-day symposium is a unique forum for trainers and educators to discuss their needs and ideas for incorporating object technology into training plans and courses under consideration of the human factor. We will celebrate this year's symposium by reflecting on what we achieved in the last decade and also focusing on the challenges we face in the near future.

The Educators' Symposium facilitates the swapping of ideas in a number of ways, including featured talks by professionals on the cutting edge of OO education, paper presentations, activity sessions, posters, demonstrations and other opportunities to exchange course materials. All attendees are invited to actively participate in the symposium by bringing any course related material they would like to share (exercises, teaching and learning tools, effective evaluation questions and methods, URLs, descriptions of course needs, things you want to brag about or advertise, etc.).

8:30 am - 9:40 am

Marriott Hotel — Salon C-D
Program

Welcome and DesignFest Introduction

Jutta Eckstein,
Objects in Action

Ralph Johnson,
University of Illinois at Urbana-Champaign

Keynote

Object Technology Education Today

Linda Northrop,
Software Engineering Institute

Over the last ten years, object technologists have spawned major developments that have changed our implementations and our approach. Use cases, standardized middleware, Java, UML, aspect-oriented programming, and design patterns have influenced the way we build systems and have now become part of the technical fiber we routinely use. What about object technology education? Has it kept pace? Is the focus properly fixed? Together we will examine the ten year journey and the challenges that remain.

9:40 am - 10:00 am

Panel

Educators’ Symposium: The First Ten Years

Mary Lynn Manns,
University of North Carolina at Asheville

Ed Gehringer,
North Carolina State University

Joe Bergin,
Pace University

Rick Mercer,
University of Arizona

This year marks the 10th consecutive OOPLSA Educators’ Symposium. In the beginning, Educators’ Symposia were organized rather like technical conferences, with invited talks and contributed papers. More recently, they have evolved to a participatory format, with poster and demonstration sessions, and design exercises. The panelists, who have attended all ten symposia, will reminisce on these symposia by considering the topics, the opinions, the debates, the people, and the hype. We will connect with happenings in other parts of the OOPSLA conference and the trends of OO education in industry and academia.

10:00 am - 10:30 am

Break and Sneak-through the DesignFest

10:30 am - 11:30 am

Paper Presentations: Rethinking — How do we teach and what?

Extreme Programming in an Introductory Course Taught using the Java Programming Language

Daniel Steinberg,
Dim Sum Thinking Inc.

Daniel Palmer,
John Carroll University

It seems natural to introduce Extreme Programming (XP) to an upper level course in software engineering. This methodology is well suited to working on projects for real clients that have restrictive time constraints for completing a project that satisfies the client’s most pressing needs. In this paper, we’ll discuss the benefits of introducing at least some of the practices of XP much earlier in the curriculum.

On the Conflict between Teaching Software Engineering and Teaching Computer Science

Michael Whitelaw,
Charles Sturt University, Australia

There is no clear distinction in the computing field between the conservatism of a (Software) Engineer and the risk-taking curiosity of a (Computer) Scientist. Reflecting this confusion, most of our courses transmit ambiguous messages to students as to what is permissible and what is not permissible. Some of the differences that the contrasting attitudes take can be seen in different approaches to life-cycle methodologies, code-reuse, tools, and even the very understanding of concepts like object. At the very least, we need to clarify these differences for our students; in the long term we must expect our courses to diverge.

Containers and Iterators: An Example of Constructive Elicitation of Patterns

Arturo Sánchez,
University of North Florida

Design Patterns are currently being used as an integral part of curricula that introduces object-oriented technology from the first course and up. A popular approach to using patterns in introductory computer science courses consists of presenting a catalog to be applied for tackling problems whose complexity level is appropriate for the course in question. Although this is the natural approach for it, goes in tune with the goal of promoting a lingua franca among software designers and programmers, it is not the only viable approach. In this paper we present an example that illustrates a constructive approach to the problem of unveiling three cognitive processes that take place in the discovery of patterns, namely observation of repeatability, lifting (or generalization) of specific solutions to more general ones, and specialization of general solutions to specific ones. The example starts with a very simple and specific iteration mechanism (Java’s Enumeration) and covers a series of problems whose solutions converge to the Iterator Pattern. Our example can be used to introduce novices to the concept of patterns, and also to introduce the already initiated pattern mining.

Grasp All, Loose All

Marianna Sipos,
Budapest University of Technology and Economics, Hungary

Learning a new programming paradigm without any practical experience seems to be impossible. So it is not enough to know the OO concepts, but also a language and a framework have to be taught. This paper describes my solution, which gives the students clear concepts, success with small exercises, and the feeling that they are capable of solving bigger problems in project work.

11:30 am - 12:15 pm

Demonstration

DesignFest on Stage

Ralph Johnson,
University of Illinois at Urbana-Champaign

The OOPSLA DesignFest is about design and creativity. The DesignFest is a free event (for conference registrants) that was created to give OOPSLA attendees the opportunity to learn more about design by doing it. In the DesignFest people work in small groups to solve a particular design problem, bringing to bear their experience and skills in object-oriented design and/or experience working on similar problems. The goal is to learn new techniques from each other and to uncover and articulate the analysis and design patterns that are already used subconsciously. DesignFest is also a great way to get some first-time experience, which means it is an excellent technique for teaching Design!

12:15 pm - 1:15 pm

Marriott Hotel — Meeting Rooms 8-9
Lunch

1:15 pm - 2:00 pm

Invited Talk

Object-Oriented Thinking Is Easy To Learn, But Seldom Taught. Experiences 1965-2001

Kristen Nygaard,
University of Oslo and the Norwegian Computing Center, Norway

Kristen Nygaard invented object-oriented programming together with Ole-Johan Dahl in the 1960s. The languages Simula I and Simula 67 introduced objects, classes, inheritance, virtual quantities and quasi-parallel (multi-threaded) sequencing. He is of the opinion that the standard pedagogic in teaching object-oriented programming is wrong. His approach, which he has lectured around the world, is that one must start with “sufficiently complex examples” instead of “sufficiently simple examples” in order to teach the pupils the “world view” of object-orientation. Otherwise the pupils will continue programming as before, albeit in an object-oriented language.

2:00 pm - 3:00 pm

Paper Presentation: Teaching Collaboration Skills

Redesigning CS101: A Learning Based Approach

Alfonso Rodriguez and Ariel Ortiz,
ITESM Campus, Mexico

Nowadays, software developers are required to build better systems in less time. We expect computer professionals to work in teams, learn new technologies, solve tough problems, and exceed all past achievements. But as educators, are we really promoting these skills and values? This paper presents the experiences obtained from redesigning an objects-first CS101 course, in which students were made responsible of their own learning practice. Interesting projects developed in teams and built from self-acquired knowledge are the foundation of our proposed scheme.

Techniques for Active Learning of OO Development

Robert Biddle, James Noble and Ewan Tempero,
Victoria University of Wellington, New Zealand

We describe our use of active learning techniques to teach OO development. We have developed new techniques, adapted from CRC cards, to teach use cases for requirements gathering. We have also adapted CRC cards to teach the principles of OO. Our approach has been tried with large university classes as well as industry groups, programmers as well as business analysts and managers.

Cooperatively Enriching Education: Industrial Projects for Academic Credit

Dr. Edward F. Gehringer,
North Carolina State University

Dave Maeda,
IBM VisualAge for Smalltalk Group

Industry and universities have long worked together on cooperative education (co-op) projects in which a student spends a semester or more working in industry. Could even shorter-term projects be of mutual benefit? During the Spring 2001 semester, NCSU and the IBM Smalltalk Group set up a program where an NCSU faculty member and IBM employees jointly supervised students working on small, well-defined Smalltalk projects for academic credit. Benefits to the students included gaining experience on a real-world software project directly related to their coursework, and making valuable contacts in industry. Benefits to the NCSU Computer Science department included offering its students the ability to work with practicing software developers in a small-group setting. For its part, IBM obtained short-term help on three projects, and acquaintance with several potential job candidates.

3:00 pm - 3:45 pm

Workshop Reports, Poster Session and Break

3:45 pm - 5:15 pm

Activity

Looking for Abstractions in a Concrete World: Candidates, Responsibilities, and Collaborations

Rebecca Wirfs-Brock and Alan McKean,
Wirfs-Brock Associates

CRC has traditionally stood for “Classes-Responsibilities-Collaborators.” But Classes are too concrete. Jumping to classes too soon results in a lifeless design. We should be thinking more abstractly: in terms of distinct roles that collaborate. Classes imply code; roles imply behavior. Thinking about roles can simplify a design, but it is hard to make the leap from concrete objects to abstract roles. This talk demonstrates how to think about candidate roles and their responsibilities, what the benefits are, and how to implement these abstractions with interfaces and classes.

5:15 pm - 5:30 pm

Wrap-up and Open Microphone