Object Principles: Back to Basics

John Daniels
Syntropy Ltd
jd@syntropy.co.uk

Jutta Eckstein
Objects in Action
jeckstein@acm.org

Abstract. We present the results from an Educators' Symposium session. As OT becomes increasingly submerged beneath delivery technologies (like components), it's ever more important that educators are clear about the object principles that provide the benefits we claim. Will we be able to recognize the important principles in the diverse technologies?

This is the result of an Activity Session, which was part of the Educators' Symposium. In this session the participants of the Symposium actively discussed the topic described below. This report presents the agreed positions and statements.

As object technology becomes increasingly submerged beneath delivery technologies (such as components) that exploit its potential, it is ever more important that educators are clear about the underlying object principles that provide the benefits we claim.

This session attempted to answer the question: What is OO really all about? Can you write down and explain the key principles that make the object-oriented approach so important and powerful? It's surprisingly difficult to do this. Worse still, the concepts we think are "essential" to OO seem to change over time. For example, a few years ago the concept of class inheritance was thought to be essential. Now we see that class inheritance is an implementation code-sharing trick, and even interface inheritance is but one way - out of many - of describing conformance to contracts.

We tried to identify the important principles in the diverse technologies that claim to be "object-oriented."

The session was highly interactive, in the "think tank" style. We started work in small groups, and determined an agreed position. Groups then gradually merged, ending with four large groups, trying to build a consensus as we went along. The session's result was a number of position statements that we present here.

Top 5 important principles:

Group I:

- Having fun because it's maintained (? ed.)
- Aesthetically clean encapsulated interface
- Reasonable economics via potential reuse
- Parameterizing changes via encapsulation
- Putative real-world mobile objects that bind late

Group II

- Encapsulation
- Polymorphism
- Enables easy and naive behavioral design
- Enables piecemeal development
- Design by contract

Group III

- Encapsulation
- Abstraction
- Dynamic Binding
- Inheritance
- Buzz words

Group IV

- Objects to model things
- Encapsulation of data ≠ functions
- Separation of interface and implementation
- Polymorphism for obtaining meaning from context type inheritance

OOPSLA 2000 Companion Minneapolis, Minnesota