PANEL: DESIGN METHODOLOGY FOR OBJECT-ORIENTED PROGRAMMING[1]

Abstract:

This panel is aimed at software developers who use object-oriented programming techniques. Some developers opt for object-oriented programming in search of enhanced customizability, extensibility, and reusability of their software. Others opt for it because of its perceived utility for modeling and simulation, for exploratory programming, or for user interface development.

Whatever the reason, the prospective object-oriented programmer is faced with a programming style that differs significantly from the procedural style in which most programmers have been raised. Along with the new style, a new set of design decisions must be made---and the programmer has little intuition for how to make them. For example, the decisions include:

- *What object-oriented language is appropriate for the application?* … how to select from the available languages according to object-oriented features (e.g., multiple inheritance), runtime efficiency, and development environment.
- *What's an object?* … how to select a level of granularity appropriate to the application.
- *What is the right set of classes and how do the classes go together?* … many different hierarchical decompositions can be made, according to a variety of inter-object relationships, like taxonomic [is-a or subclass], part/whole, or other application-specific relationships.
- *What is an appropriate set of generic methods and in which classes should these methods be placed?* … in order to achieve best use of inheritance, to maximize clarity, or to maximize reuse.
- *What is the impact of the development environment on the design of object-oriented programs?* … would Smalltalk programs be different if written in a text-only environment.
- *Are development methodologies devised for procedural languages (e.g., functional or structural decomposition, stepwise refinement, rapid prototyping) appropriate for object-oriented languages?*} … and what to do if and when they are not.
- *Is Fred Brooks' Mythical Man-Month thesis relevant to object-oriented programming projects?*

The panelists represent a number of different communities experienced in object-oriented programming, including software engineering, simulation, and artificial intelligence.

Each panelist will present a very brief *position paper* on design methodology appropriate to programming in an object-oriented style. The panelists, together with the audience, will then discuss and debate the issues. The focus will be on practical issues involved in the *use* of object-oriented techniques, as opposed to theoretical issues involved in the design of object-oriented languages.

Moderator: Reid Smith [Schlumberger Palo Alto Research]

[1] Panel on Design Methodology for Object-Oriented Programming. *Conference on Object Oriented Programming Systems Languages and Applications (OOPSLA'87),* Orlando, FL, USA, October, 1987.

DESIGN METHODOLOGY FOR OBJECT-ORIENTED PROGRAMMING

WARD CUNNINGHAM
Tektronix , Inc.

NORM KERTH
University of Portland

GREGOR KICZALES
Xerox Palo Alto Research Center

BERTRAND MEYER
Interactive Software Engineering, Inc.

NORM MEYROWITZ
Institute for Research in Information
Brown University

REID SMITH
Schlumberger Palo Alto Research

AIMS

DEVELOP INTUITIONS ABOUT DESIGN, IMPLEMENTATION, & DEVELOPMENT DECISIONS THAT ARISE IN OBJECT-ORIENTED PROGRAMMING

"ENGINEERING AESTHETICS"

CRITERIA FOR EVALUATING THESE DECISIONS

- **EXPERIENCE**
  - **WHAT WORKS?**
  - **WHAT DOESN'T?**

DIMENSIONS

- **GRANULARITY**

  ... WHAT'S AN OBJECT?

- **CLASSES / METHODS**

  ... WHAT ARE THE 'RIGHT' ONES?

- **OO LANGUAGES**

  ... WHICH FEATURES MATTER?

- **DEVELOPMENT ENVIRONMENT**

  ... WHAT IS ITS EFFECT ON PRODUCTIVITY / QUALITY?

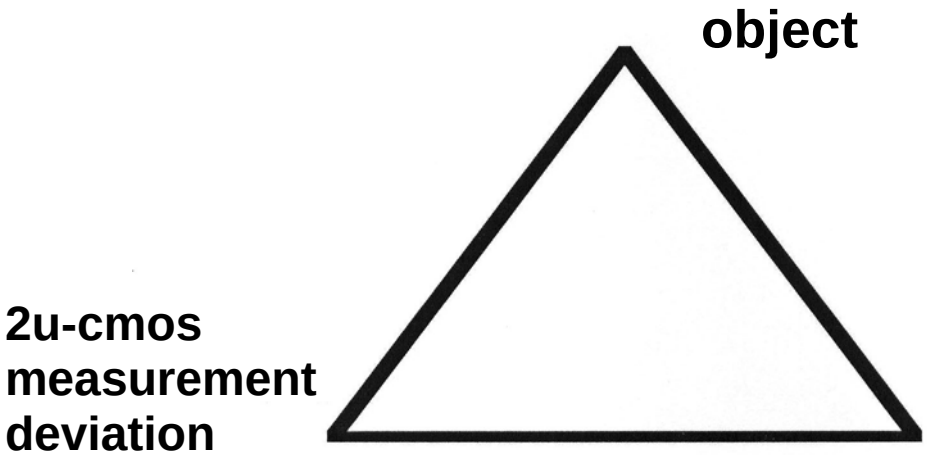
- **DEVELOPMENT METHODOLOGY**

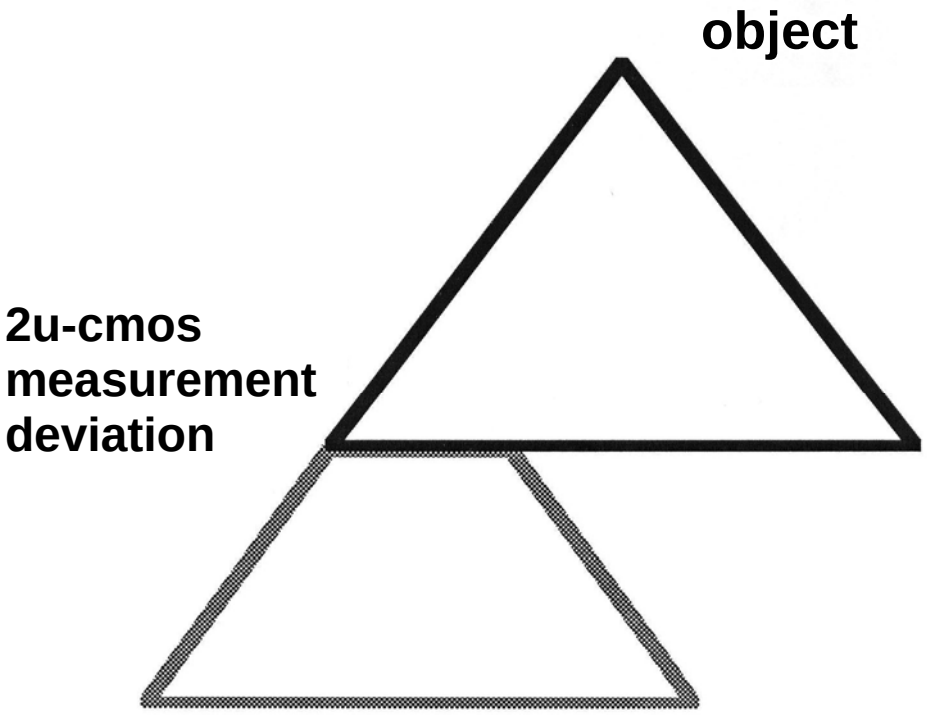
  ... CAN THOSE DEVELOPED FOR PROCEDURAL LANGUAGES BE IMPORTED?

Two additional slides – not used during panel.

Extending An Object-Oriented Framework

Theory

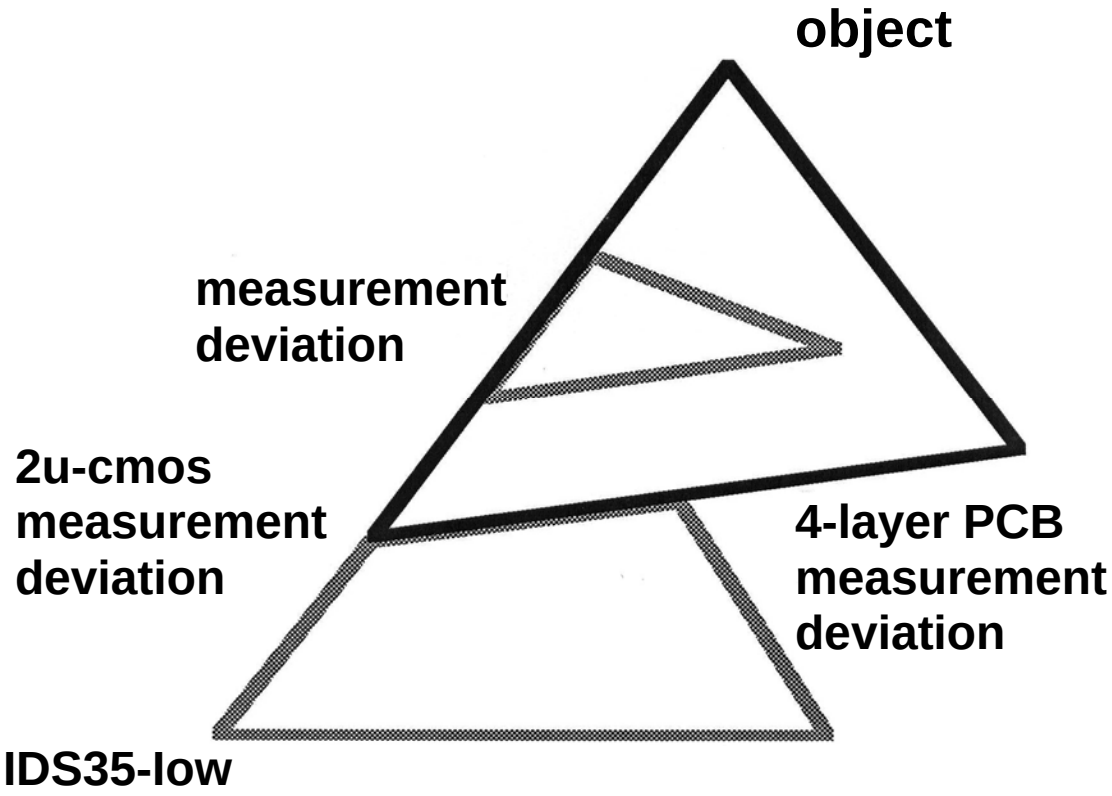




IDS35-low

Extending
An
Object-Oriented Framework

Actuality



Moral: Extension of an object-oriented framework is done by specializing <u>and</u> by generalizing.