"Enhancing Reuse with Information Hiding"

E.T. Proceedings of the Workshop on
Reusability in Programming, 1982
Reprinted in Software Reusability,
Volume I, Concepts and Models, 1989

Presented by:
Brian Kellogg and Michelle Pirtle

Reuse with Information Hiding:
Overview

- Reasons code is not reused:
  - Code complexity
  - Cheaper cost to develop new code
  - Lack of knowledge about existing code
  - Existing code is too general

- Objective: Demonstrate how the programming technique of information hiding can overcome the above reasons for not implementing software reuse.

(Parais, 1985)

Reuse with Information Hiding
(Continued)

- Definition of information hiding:
  - Aspects of a module that can change independently should be secrets from other modules; an interface contains the assumptions.
  - Parais' view of a module using Information Hiding
    - Role of modules
    - Secrets of modules
    - Facilities of modules
  - Two types of secrets
    - Primary secret
    - Secondary secret

(Parais, 1985)
Reuse with Information Hiding: Overview

- Reasons code is not reused:
  - Code complexity
  - Cheaper cost to develop new code
  - Lack of knowledge about existing code
  - Existing code is too general
- Objective: Demonstrate how the programming technique of information hiding can overcome the above reasons for not implementing software reuse.

Reuse with Information Hiding Impact

- Example
  - Naval Research Laboratory A->E flight software
- Impact
  - Object-Oriented programming
  - Decomposition of program into objects
  - Hierarchy of program
  - Interaction between objects

Empirical Investigation on the Innovation Diffusion in a Software Development Process

- Objective: To observe the behaviors and achievements of programmers after learning a new programming technique.
- The Experiment
  - Subjects
  - Tests: 5, one-week tasks
  - Design methods subjects could choose from

(Partan, 1985)
Innovation Diffusion (Continued)
- Results of the experiment
  - Skills need to be acquired through experience, training, and guidance
  - Deliverables and wasted effort were equal between subjects
  - Programmers will modify design process to fit their understanding and knowledge
  - Information Hiding is more difficult to learn than 5th Normal Form
- Use of Parnas' work
  - Testing implementation of information hiding
  - Demonstrated the difficulty of good information hiding
  - Stated designers are better off to design own method of information hiding

Assuring Good Style for Object-Oriented Programs
- Lieberherr and Holland, 1989, IEEE Software
- Objective: To improve encapsulation and modularity with a new programming rule called "Law of Demeter".
- Definitions:
  - Supplier object/class
  - Preferred supplier object/class
- Three Forms of the Law of Demeter
  - Object Form
  - Strict Class Form
  - Minimization Class Form

Style for OO Programs (Continued)
- Principles the Law uses
  - Coupling
  - Information hiding
  - Information restriction
  - Localization of information
  - Structural induction
- Benefits of implementing the Law
  - Maintainability
    - Limits hierarchy structure enforces
    - Ease of maintenance
  - Understandability
    - Use of modularity
    - Smaller modules
    - Predictable message-passing patterns
Style for OO Programs (Continued)

- Example
  - Demeter System™, Demeter Research Group at Northeastern University

- Extension of Parnas' work
  - Different method to achieve the same goals of maintainability and understandability
  - Implements Parnas' work on information hiding and extends this method to include all classes (assumes all classes may be modified)
  - Differentiates between information hiding and information restriction

[Libermann and Eick, 1999]

Reuse Approaches in Software Engineering and Knowledge Engineering: A Comparison

- Thomas Hemmann, 2nd International Workshop of S.E.
- Objective: Illustrate ways that reuse in software engineering can be applied to the field of knowledge engineering.
- Reuse methods found in Software Engineering:
  - Modularity and Information Hiding
  - Commonality
  - Maintainability
  - Portability
  - Evolution
  - Software Libraries

[Hemmann, 1993]

Knowledge Engineering (Continued)

- Extends Parnas' challenges to Software Engineering
  - Reused code may contain bugs.
  - Social-economic concerns
  - Not-invented-here syndrome
  - Lack of incentive to implement reuse
  - Managers lack experience in reuse
  - High-initial cost
- Many examples of software reuse exist.
- Knowledge Engineering
  - Knowledge: facts, beliefs, heuristics
  - Knowledge-based systems, expert systems

[Hemmann, 1993]
Knowledge Engineering (Continued)

- Reuse in Knowledge Engineering: proposed methods
  - Knowledge Query and Manipulation Language
    - Provides common protocol for AI-systems to communicate.
  - Knowledge Representation System
    - Creates standard features of a language to avoid translation.
  - Reusable, domain specific ontology
    - Ontology: "An explicit formal specification of how to represent the objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them" (dictionary.com)
  - Problem-solving methods
    - Defining basic methods that can solve generic tasks across domains.
  - Knowledge Interchange Format
    - Translate one language to another language.

Object-Oriented Documentation

- Johannes Sametinger, Journal of Computer Documentation
- Objective: To enhance the reusability of documentation by applying techniques found in object-oriented programming.
- Distinguishes between users and those responsible for maintaining/updating a class.
- Documentation can not be contained in one document, six aspects:
  - Static overview
  - Dynamic overview
  - Class interface description
  - Task interface description
  - Class implementation description
  - Task implementation description

OO Documentation (Continued)

- OO techniques applied to documentation:
  - Inheritance
    - Decompose document into sections: name, summary description, use, variable descriptions, method descriptions.
    - Sections can be reused, removed, replaced, or extended.
  - Information Hiding
    - Distinguishes between public, private, and protected parts of the documentation.
    - Only displays documentation relevant to the user.

(Sametinger, 1994)
OO Documentation (Continued)

- Introduces a tool named D0gMA
  - Has the ability to browse classes and their documentation.
  - Applied documentation using D0gMA to a framework ET++.
- This paper extends Parnas' work by extending information hiding into documentation.