

Software Reuse

Software Engineering Course
Given by: Arnon Netzer

What is reuse?

To use parts of one product in another product
with a different function.

What can be reused?

code, documentation, design, testing,
anything

Is porting an example of reuse?

NO!!

How Reuse Happens?

- *accidental reuse*:
when a part turns out to be reusable by accident.
- *deliberate reuse*:
when a part is purposely built for possible reuse.

Obvious Reuse Examples:

- Cut & Paste (*accidental reuse*).
- *Visual Basic* - a system that generates code automatically using built-in reusable modules.
- *stdio.h* - subroutine libraries are bunches of reusable modules.

Advanced Reuse Examples:

- Hopefully - An organization saves and reuses code modules it has developed itself .
- Utopically - Organizations sell each other modules for reuse.

Purpose of Reuse

- Cheaper products
- Better quality products

Cheaper products ?!!

- Shorter development time.
- Increase productivity.
- Decrease testing domain.
- Easier maintenance.

Better quality products !!?

Code that was written for reuse:

- Has better specifications.
- Is more thoroughly tested.

Software utilizing reused code has to be:

- Better specified.
- Well standardized.
- Organized.



Drawbacks of Reuse

- Code designed for reuse is more expensive to create.
- Integrating reusable code introduces an overhead in the development process.
- Maintaining the reusable resources requires added mechanisms.



How to Create Reusable Code

Problem Domain Vs. Solution Domain

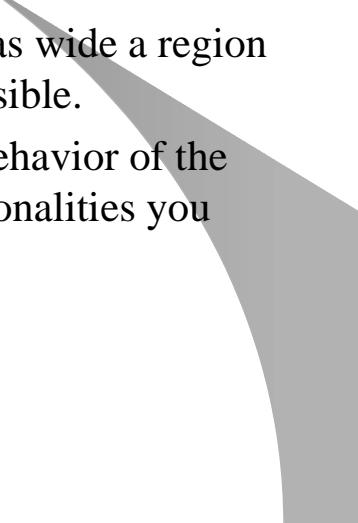
- Designs which attempt to adhere as closely as possible to the structure of the problem (application) domain are said to be application-driven.
- Designs which attempt to simplify the solution are said to be solution-driven.

Problem Domain Design

- Design that well describes one application space, should be able to fit a similar application space with minimal design adjustment.
- Small and localized change in the original problem specification should result in a small and localized change in the object-oriented design.



Spanning the Problem Domain

- Design software that spans as wide a region of the problem space as possible.
 - Regard the properties and behavior of the object rather than the functionalities you currently need.
- 



How to Use Reusable Code

Inheritance

```
class GeometricObject {
    Point position;    // center of object
    Point bbox;       // bounding box
    angle orientation; // orientation
public:
    void move(Point r); // abs. translate
    void rotate(double angle); // abs. angle
    virtual void draw(); // display on screen
    // etc...
};

class Rectangle : public GeometricObject {
    // etc...
};
```

Composition

Composition is the process of composing a class object out of several object of other related classes.

```
class Stack {
    List list;
public:
    int push(char *s) {
        return list.add(s);
    }
    char *pop() {
        return list.removeAt(List::First);
    }
    int size() {
        return list.size();
    }
};
```


Effects of Reuse on Quality Productivity and Economics

Metrics collected on two case studies
at Hewlett-Packard.
Wayne C. Lim 1994.

1st case study-Overview

- The study was done in the Manufacturing Productivity section of HP's Software Technology Division.
- The MP section produces larg-application software for manufacturing resource planning.
- The study was started in 1983.

1st case study-Technical Data

- Reuse was done on application source code etc.
- Total reusable code size was 55,000 lines of noncomment source statements.
- The code was written in Pascal and SPL.

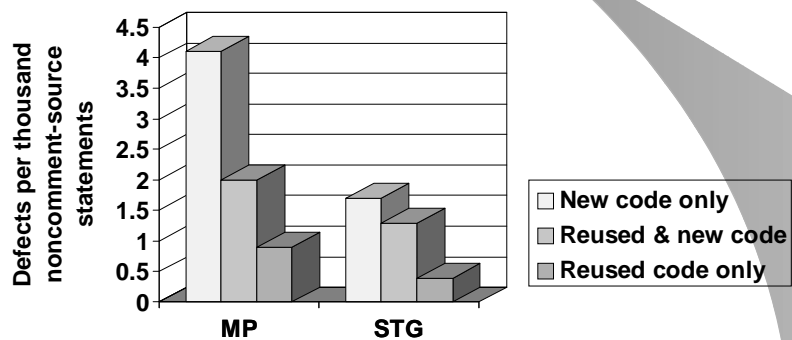
2nd case study-Overview

- The study was done in the San Diego Technical Graphics Division of HP.
- The STG develops applications for plotters and printers.
- The study was started in 1987.

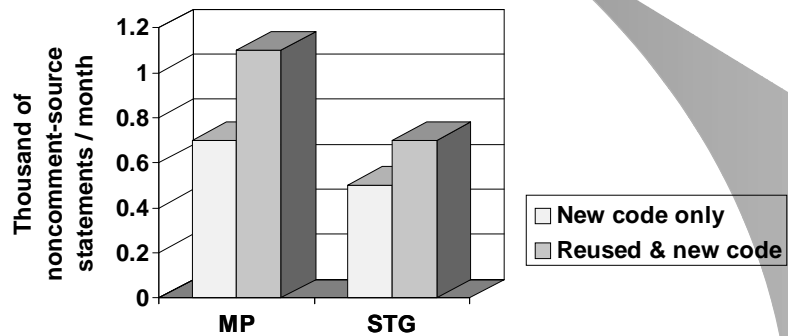
2nd case study-Technical Data

- Reuse was done on application source code etc.
- Total reusable code size was 20,000 lines of noncomment source statements.
- The code was written in C.

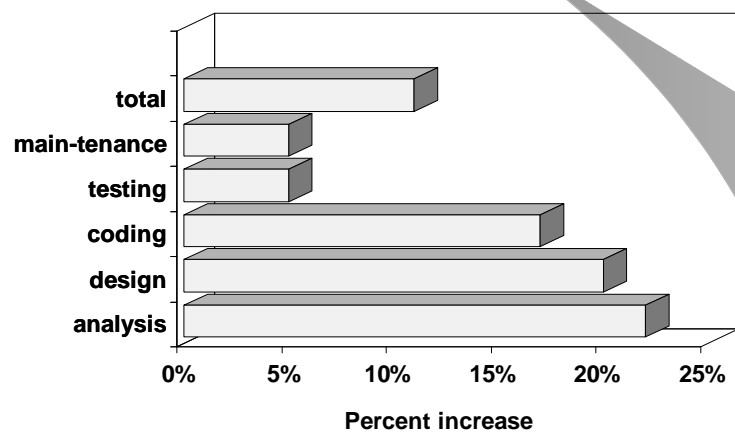
Code Quality



Coding Productivity



Additional Effort in Creating Reusable Code in STG



Reuse Case Study - Economics

	MP	STG
time horizon	1983-1992	1987-1994
start-up resources required	26 engineering months \$0.3 million	107 engineering months \$1.4 million
Ongoing resources required	54 engineering months \$0.7 million	99 engineering months \$1.2 million
Gross cost	80 engineering months \$1.0 million	206 engineering months \$2.6 million
Gross savings	328 engineering months \$4.1 million	446 engineering months \$5.6 million
Return on investment	410%	216%
Break even year	2 nd year	6 th year