THE EVIDENCE FOR DESIGN PATTERNS


Prepared by
Anastasiia Onchukova

Tartu, Spring 2012
Content

• Introduction
• 7 Design Patterns
• Experiment idea
• Experiment 1: Pattern Documentation
• Experiment 2: Pattern <> Simple program
• Experiment 3: Team Communication
• Results
• Conclusions
Introduction

• Became popular in 1995 year

• Patterns describe interaction between components, which should be adopted for circumstances.

• Advantages:
  • Higher software quality
  • Greater programmer productivity
  • Better team cohesion
  • Improved design skills for inexperienced programmers
7 Design Patterns

- **Abstract Factory** - creating sets of matched objects
- **Bridge** - handling different platforms
- **Composite** - construct part hierarchies
- **Decorator** - adding information to existing class
- **Observer** - changes of data structures
- **Template Method** - build extensive software
- **Visitor** - methods to walk recursive data structure
Chunking theory

• “Chunk”- unit of information

• Short-term memory:
  + Fast and accurate
  - Limited capacity (7±2 “chunks”)

• Patterns make units that groups several elements together.
Experiment idea

- 1996- series of experiment

- The experiments need to test programmers performance when working with sets of classes or other software elements occurring in design patterns.
Experiment 1: Testing Pattern Documentation

- January 1996, University of Karlsruhe
- 64 graduate, 10 undergraduate students
- Solutions on paper using Java
- 2 programs similar in size and complexity
Experiment 1: Testing Pattern Documentation

• Hypothesis 1
  • Documentation of patterns **speed up** pattern relevant maintenance tasks

• Hypothesis 2
  • Documentation of design patterns **reduces errors** in pattern relevant maintenance tasks
Experiment 1: Testing Pattern Documentation

- Counter-balanced experiment design
- Phonebook (Observer & Template method)
- And-Or-Tree (Compositum & Visitor)

© “The evidence for design patterns”. Walter Tichy
Experiment 1: Results

• Without pattern documentation:
  • Slower subjects produced solutions of much lower quality.

• With pattern documentation:
  • Quality not dependent of time required.
Experiment 2: Pattern Solutions via Simple Solutions

- November 1997, Munich
- 29 professional developers
- Solutions on paper, unlimited time
- 4 programs
- 2 versions of each program: program with one or two patterns & functionally equivalent but with simpler structure instead of pattern
Experiment 2: Pattern Solutions via Simple Solutions

© “The evidence for design patterns”. Walter Tichy
Experiment 2: Conclusions

- Pre-test:
  - The pattern solution took longer to modify than the alternative solution.

- Post-test:
  - The alternative version still takes about the same time, but now the pattern solution is completed more quickly.

- When choosing between patterns and simpler, less flexible alternatives, it is wiser to choose the patterns
Experiment 3: Patterns & Team Communication

- April 1999

- Record interaction in the group during design maintenance tasks.

- Group: expert & novice

- Explanation phase, pattern course, post-test
Experiment 3: Results

- In the pre-test, the expert dominates the conversation throughout, but in the post-test, the conversation becomes balanced.

- Team members can communicate more effectively with design pattern knowledge.
Conclusions

• 3 experiments, 2 of which were replicated, gathered evidence about the effectiveness of software design patterns

• Open questions to be tested:
  • Cost efficiency of using patterns
  • Efficiency of using patterns during initial development
Thank you for attention!