Welcome to SENG 371 Software Evolution Spring 2013

A Core Course of the BSEng Program

Hausi A. Müller, PhD PEng
Professor, Department of Computer Science
Associate Dean Research, Faculty of Engineering
University of Victoria

Announcements

- Lab attendance
  - Has been a problem as of late — needs to change
  - Several questions on labs on final exam
- Final exam
  - Sat, April 13 — 7:00 -10:00 pm
- Teaching evaluations
  - Next week
- Marking
  - A2 should be graded this week
  - Midterm and A1 graded
  - Marks posted
- Course website
  - http://www.engr.uvic.ca/~seng371
  - Lecture notes posted
  - Lab slides and activities are posted
- Assignment 3
  - Due Thu, April 4
  - Part I — Define software evolution terms
  - Part II — Investigate two AntiPatterns — Vendor-Lock-In — Analysis Paralysis
  - Part III — Refactoring in IBM Eclipse and MS Visual Studio and Blob AntiPattern
  - Cite your sources
  - Submit by e-mail to seng371@uvic.ca

2

Reading Assignment

- Murphy, Notkin, Lan: An empirical study of static call graph extractors, *ACM Transactions on Software Engineering and Methodology (TOSEM)* 7(2):158-191 (1998)
  - http://dl.acm.org/citation.cfm?id=279314
- Müller, Jahnke, Smith, Storey, Tilley, Wong: Reverse Engineering: A Roadmap, in *The Future of Software Engineering*, pp. 47-60 (2000)
  - http://dl.acm.org/citation.cfm?id=336526
- Storey: Theories, tools and research methods in program comprehension: past, present and future, *Software Quality Journal* 14:187-208 (2006)
  - http://webhome.cs.uvic.ca/~chisel/pubs/storey-pc-journal.pdf
- Brown, Malveau, McCormick III, Mowbray: *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis*, John Wiley (1998)
- AntiPatterns Tutorial and Website
  - http://www.antipatterns.com/briefing/index.htm
  - http://www.antipatterns.com

3

Software AntiPatterns

http://en.wikipedia.org/wiki/The_Comedy_of_Errors

Final Exam Questions

- How can you turn an AntiPattern into a good solution?
- Describe the "Vendor-Lock-in" AntiPattern
- What are the main causes for AntiPatterns?
- What are the differences between Development, Architecture, and Management AntiPatterns?
- How can a design pattern evolve into an AntiPattern?

5

Final Exam Questions …

- What are the symptoms or how can you recognize the "Design by Committee" AntiPattern?
- How are the "Vendor Lock-in" AntiPattern and levels of indirection related?
- During software maintenance "analysis paralysis" can occur. Describe this phenomenon.
- Why is it useful for a software architect to study AntiPatterns?

6

Overview

- Motivation
- Reference model
- Software Development AntiPatterns
- Software Architecture AntiPatterns
- Software Management AntiPatterns
- Summary

References

- Brown, Malveau, McCormick III, Mowbray
  AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, John Wiley & Sons, 1998
- AntiPatterns Tutorial
  by McCormick III, Mitre Corp.
  - http://www.antipatterns.com/briefing/index.htm
- AntiPatterns web site
  - http://www.antipatterns.com/
- Anti Patterns catalog
  - http://c2.com/cgi/wiki?AntiPatternsCatalog

http://www.antipatterns.com

AntiPatterns

Welcome to the Hot best sellers,

Click here to order **AntiPatterns** or **Anti-Patterns and Patterns in CM** right now from ...

amazon.com
Anti Patterns
Anti-Patterns and Patterns in Software Configuration Management
AntiPatterns in Project Management
J2EE AntiPatterns

Gain insight and understanding as you learn from this very entertaining book exactly how to recognize and then master your Software Development troubles ...

You always knew you'd laugh at your CM troubles someday! Now you can Identify, Understand and Fix your CM problems as you do so ...

Now what?! Maybe it's the Domino Effect AntiPattern. Better look here to learn what to do ...

Guide to recognizing AntiPatterns in J2EE: bad habits of code and design.

The Origins: Design Patterns

- Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley (1994). The Gang of Four Book.
- Creational Patterns
  - Singleton, factory, builder, …
- Structural Patterns
  - Adapter, composite, façade, …
- Behavioral Patterns
  - Visitor, observer, iterator, …

Origins of AntiPatterns

- The majority of published works in software sciences have focused on positive and constructive solutions
- AntiPatterns are derived by looking at the negative solutions
- Def. An AntiPattern describes a commonly occurring solution to a problem that generates decidedly negative consequences.
- AntiPatterns are also called *Bad Smells*

Origins of AntiPatterns …

- A manager or developer
  - does not know any better
  - does not have sufficient knowledge or experience solving a particular problem
  - applied a perfectly good design pattern in the wrong context

AntiPatterns and Software Evolution

- AntiPatterns are particularly prevalent during long-term software maintenance and evolution
- A software reengineer needs to assess the presence or absence of AntiPatterns in a legacy system to be able to implement the best reengineering, maintenance or evolution strategy

13

AntiPatterns and Software Evolution

- How do you compare/evaluate software development job offers?

14

AntiPatterns and Software Evolution

- How do you compare/evaluate software development job offers

- Premise
  - Recognition of AntiPatterns will make you a better software engineer
  - Refactoring AntiPatterns present in a system and/or project will result in a better, more successful, less risky software reengineering project

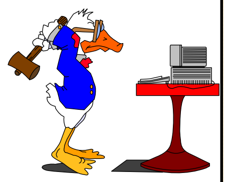
15

State of Affairs

- Five out of six software projects are considered unsuccessful
- One third of all software projects are canceled
- For delivered systems the actual budget and time is double than expected
- Silver bullets …

16

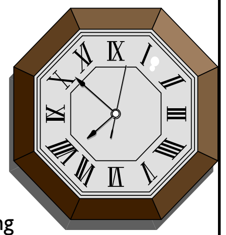
Old Silver Bullets



- Structured programming
- Top-down design
- Open systems
- Client/server architectures
- Quality code generation from models
- Object orientation
- GUI builders
- Frameworks

17

New Silver Bullets



- Component technologies
- Distributed objects
- Business objects
- Patterns
- Software reuse
- Scripting languages
- Software agents
- Network-centric computing
- Web services (SOA, Grid, Cloud)
- XML
- Extreme Programming
- Refactoring

18

AntiPattern Description Structure

- Description of the general form
- Symptoms on how to recognize the general form
- Causes that led to the general form
- Consequences of the general form
- Refactored solution on how to change the AntiPattern into a healthier situation

19

AntiPatterns Purpose

- A method for efficiently mapping a general situation to a specific class of solutions
- Provide real-world experience in recognizing recurring problems in the software industry and provide a detailed remedy for the most common predicaments
- Provide a common vocabulary for identifying problems and discussing solutions

20

AntiPattern Categories

- Development AntiPatterns
- Architectural AntiPatterns
- Management AntiPatterns

- AntiPatterns apply to software construction as well as software evolution

- Anti Patterns catalog
  - http://c2.com/cgi/wiki?AntiPatternsCatalog

21

AntiPattern Lava Flow
A first example

- Problem
  - Dead-code and forgotten design information is frozen in an ever-changing design
  - Oh that! Well Ray and Emil (they're no longer with the company) wrote that routine back when Jim (who left last month) was trying a workaround for Irene's input processing code (she's in another department now).

22

Lava Flow …

- Problem
  - Lead engineer left
  - New lead had better approach but was nervous about deleting stuff until he was more familiar with the code
  - Each volcanic eruption leaves lava streams
    - DDE leveraged
    - OLE1, OLE2
    - Support for JavaBeans
    - Support for mobile devices

Lava Flow …

- Causes
  - R&D code moved to production with CM
  - Uncontrolled distribution of unfinished or unpolished code
  - Trial approaches have not been eliminated from the code
  - Architectural scars due to old middleware

24

Lava Flow …

- Solution
  - Configuration management system which identifies and eliminates dead code
  - Evolve or refactor design
  - Sound architecture review must proceed production code development
  - Establish stable system level interfaces

Swiss Army Knife or Kitchen Sink

- Problem
  - Excessively complex class interface
  - Designer attempts to provide for all possible uses of the class
  - Complicated interface
  - Many overloaded names
  - Excessive regression test suites
  - Several Swiss Army Knifes in a single design

Swiss Army Knife or Kitchen Sink

- Refactored solution
  - Provide guidelines for using complicated standards or interfaces
  - Provide a template for exception handling
  - Contract interfaces

Group Assignment
An AntiPattern "Comedy of Errors" (Play)

- Groups of 4 students
- Pick an AntiPattern
- Develop a play to enact the AntiPattern

- Perform the play in class next week
  - Make sure all group members are involved—ideally equally
  - Include props if need be
  - Practice the play (!)
  - 5 mins for play

http://en.wikipedia.org/wiki/The_Comedy_of_Errors

Pick your play to be performed

- Reinvent the Wheel
  - Mon: Morgan, Nic, Vish, Marcelo
- Design By Committee
  - Mon: Michael, Y, Sam, Mackenzie
- Mushroom Management
  - Mon: Daniel, Brad, Dave, George
- Boat Anchor
- Stovepipe
- Architecture By Implication
- Warm Bodies
- Swiss Army Knife
- Spaghetti Code
- Blob
- Wolf Ticket
- Corncob
  - Thu: Geoff, Adam, Scott, Justin
- Golden Hammer
  - Thu: Rob, Ian, Kai, Saleh
- Walking through a Minefield
  - Thu: Jordan, Amanda, Brandon, Romil
- Poltergeists
  - Thu: Curtis, Mikko, Paul, Allan
- The Grand Old Duke of York
- Dead End
- Cut-and-Paste Programming
- Death by Planning