THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

SOFTENG 306 SOFTWARE ENGINEERING DESIGN 2 LECTURE

02 – SOFTWARE DESIGN SMELLS

Dr. Seyed Reza Shahamiri More Info

Software Design Smells*

- Design smells, the odors of rotting software, are symptoms of poor design that can be measured objectively.
- The smells are usually caused by violating design principles.
- As a designer it is your responsibility to remove smells without complicating the product.

Adopted from Agile Software Development Principles and Practices by Robert C. Martin

Software Design Smells*

Design smells are:

1. Rigidity
2. Fragility
3. Immobility
4. Viscosity
5. Needless Complexity
6. Needless Repetition
7. Opacity

Adopted from Agile Software Development Principles and Practices by Robert C. Martin

Software Design Smells*

1. Rigidity

- Rigidity smell is when the system is hard to change because every change forces many other changes to other parts of the system.
- A design is rigid if a single change causes a cascade of subsequent changes in dependent modules.
- The more modules that must be changed, the more rigid the design.

Software Design Smells*

2. Fragility

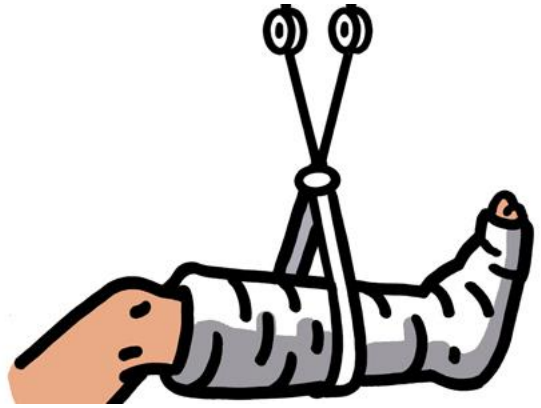
- Fragility smell is when changes to the system cause the system to break in places that have no conceptual relationship to the part that was changed.
- When fixing a problem causes more problems in areas that do not have any conceptual relationship to the code that was fixed, your software is fragile.
- Fragile software increases the likelihood of unexpected behavior and problems in your software.

Adopted from Agile Software Development Principles and Practices by Robert C. Martin

Software Design Smells*

3. Immobility

- Immobility smell is when it is hard to disentangle the system into components that can be reused in other systems.
- A design is immobile when it contains parts that could be useful in other systems, but the effort and risk involved with separating those parts from the original system are too great.
- Immobility smell affects the reusability of your software.



Adopted from Agile Software Development Principles and Practices by Robert C. Martin

Software Design Smells*

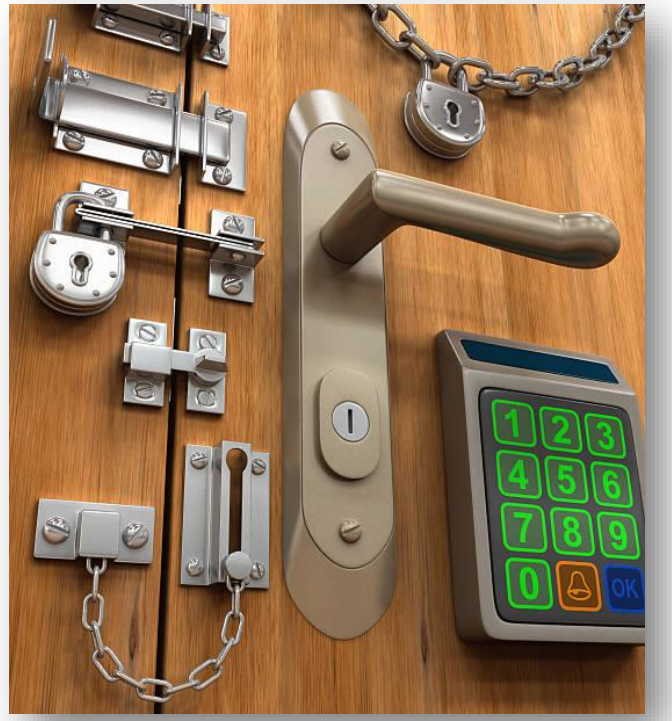
4. Viscosity

- According to Google Translate, viscosity means “the state of being thick, sticky, and semifluid in consistency, due to internal friction”.
- Viscosity smell is when doing things right is harder than doing things wrong, and it comes in two forms:
  1. Viscosity of the software: there are usually several ways to make changes to software. Some of these ways preserve the design and others not (i.e. they are hacks). Viscosity of software is high when developers find it easier to use hacks instead of design-preserve ways to change software.
  2. Viscosity of the environment is when the dev environment is slow and inefficient that encourages developers to apply hacks.

Software Design Smells*

5. Needless Complexity

- Needless Complexity smell is when the design contains infrastructure that adds no direct benefit, i.e. elements that are not currently useful.
- This frequently happens when developers anticipate changes to the requirements, and put facilities in the software to deal with those potential changes.
- By preparing for too many contingencies, the design becomes littered with constructs that are never used. Some of those preparations may pay off, but many more do not. Meanwhile the design carries the weight of these unused design elements that makes the software complex and hard to understand.



Software Design Smells*

6. Needless Repetition

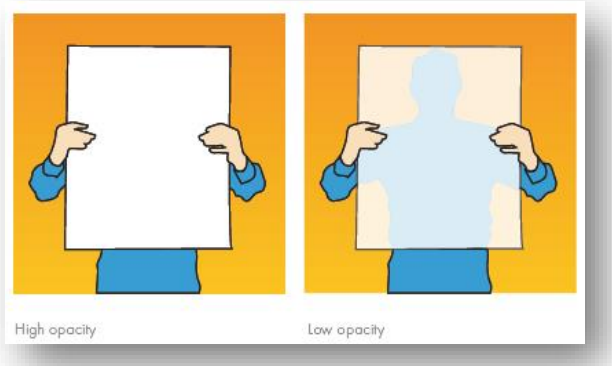
- Needless Repetition smell is when the design contains repeating structures that could be unified under a single abstraction.
- When the same code appears repeatedly, in slightly different forms, the developers are missing an abstraction.
- When changes are required, developers need to find all instances of the code in different places and make the changes one-by-one.
- This affects maintainability of the software.

Adopted from Agile Software Development Principles and Practices by Robert C. Martin

Software Design Smells*

7. Opacity

- Opacity smell is when code is hard to read and understand, convoluted, and does not express its intent well.
- Code evolves over time and tends to become more and more opaque with age.
- As software evolves, developers need to keep the code clear and expressive in order to keep opacity to a minimum.
- Opacity smell affects readability and understandability of your code.
- Principles of clean code and code reviews can help minimize opacity.



Adopted from Agile Software Development Principles and Practices by Robert C. Martin