

Refactoring Legacy Systems

Willem van den Ende – willem@livingsoftware.nl
Marc Evers - marc@piecemealgrowth.nl

© 2007 Living Software & Piecemeal Growth – Some Rights Reserved



Piecemeal Growth



Living Software B.V.

A designer knows he has achieved perfection
not when there is nothing left to add,
but when there is nothing left to take away.

Antoine de Saint-Exupéry

About Us
Design Debt Cycle
Refactoring Legacy Code
Approaches
Demo
Q&A

Who we are

- Willem van den Ende
- Refactoring
 - 1996 – New Legacy Application
 - Teams
- Independent (Living Software B.V.)
 - All-hands person
 - Software development coach
 - Trainer
 - Consultant
- blog: me.andering.com



Who we are

- Marc Evers
- Independent (Piecemeal Growth)
 - Software development coach
 - Trainer
 - Consultant
- Blog: blog.piecemealgrowth.net



What we do

More business value from software development
and
helping others do it

through



Coaching & mentoring

Training

Facilitation

Organizing conferences

Legacy Code

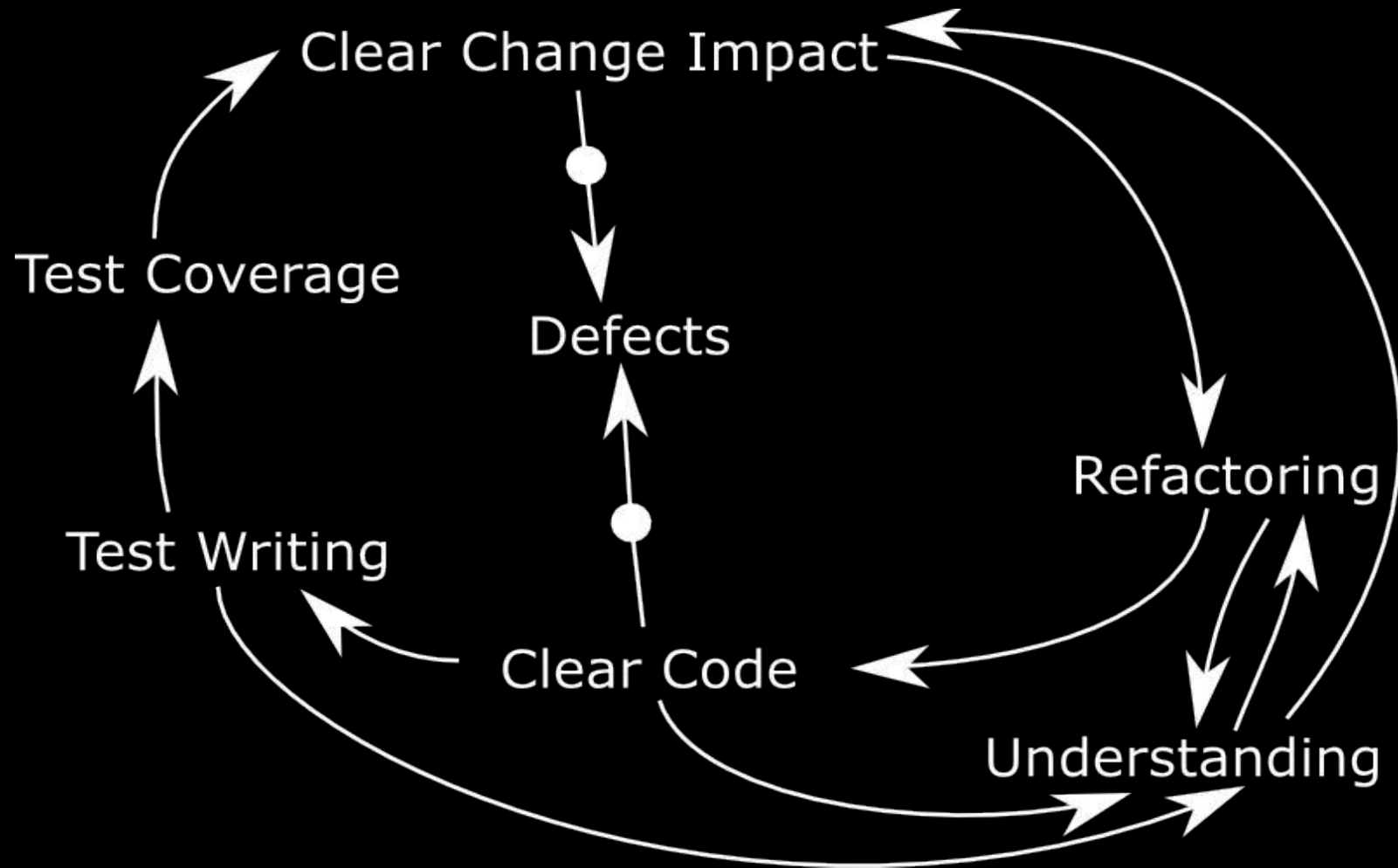


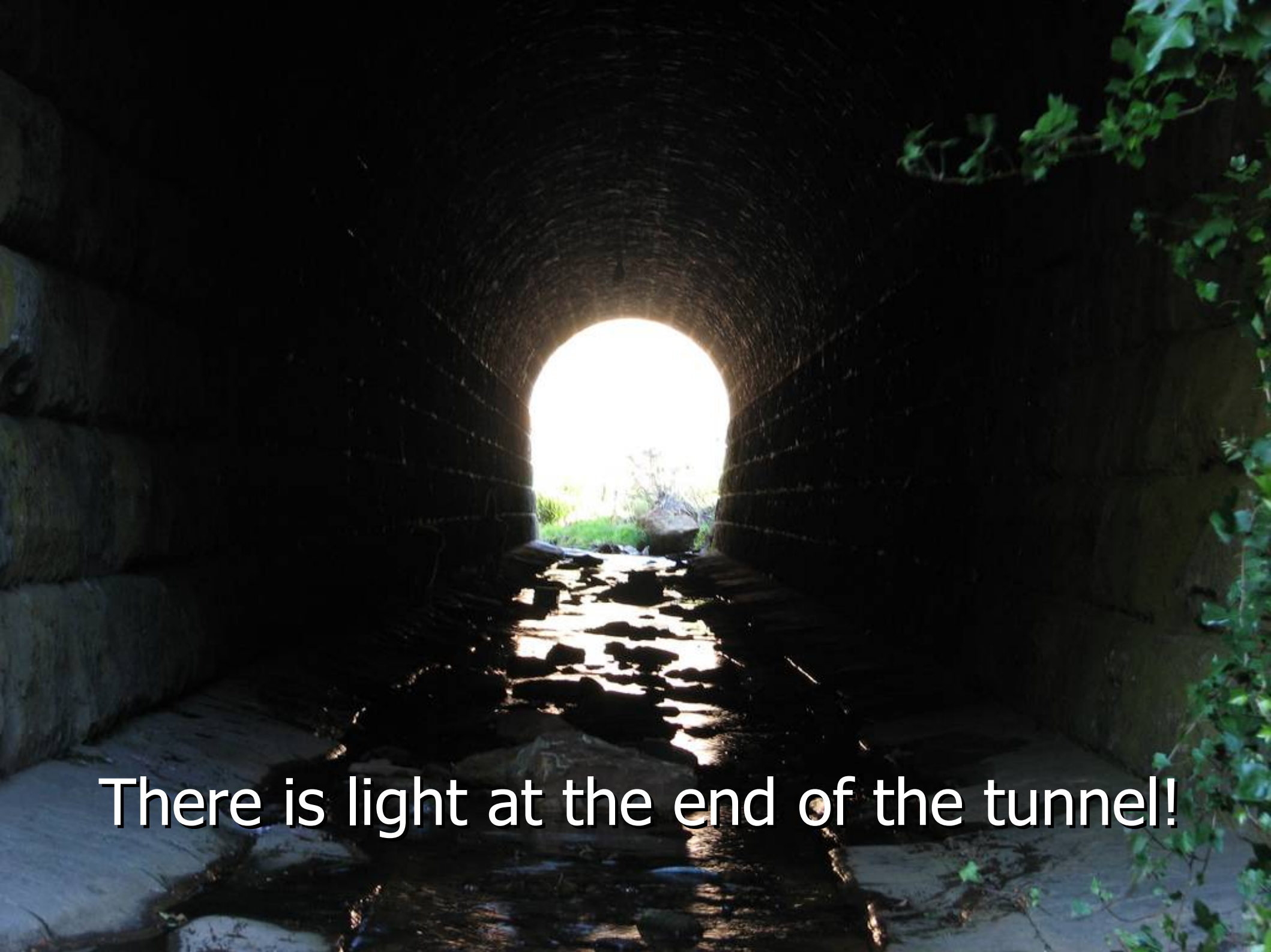
Legacy Code

A photograph of a dilapidated brick building with a roof of dark, broken tiles and overgrown green plants, symbolizing legacy code. The building is in a state of significant disrepair, with missing tiles and exposed wooden rafters. The surrounding area is overgrown with green vegetation, and the sky is overcast.

Legacy code = code without tests

Design Debt Cycle





There is light at the end of the tunnel!

Refactoring Legacy Code

A photograph of a house with its exterior walls removed, showing the internal wooden frame. The house is on a trailer, and the roof is made of red tiles. The image is used as a metaphor for refactoring legacy code, where the internal structure is improved without changing the external appearance.

Improving the internal structure of software
without changing its external behaviour

Tiny, safe, almost mechanical steps

Unit tests!

Refactoring approaches

- Cut at the seams
- Reveal intentions
- Use your tools
- One smell at a time
- Pairs & peers
- Exploratory refactoring
- Sometimes it gets worse before it gets better

Cut at the seams

- Seam = place where you can alter behaviour in a program without editing in that place
- Seam types
 - Preprocessing seam
 - Link seam
 - Object seam
- Break dependencies
 - To add tests

Object seam example

```
public class Spreadsheet {  
  
    public Spreadsheet buildSheet(Cell cell) {  
        ...  
        recalculate(cell);  
        ...  
    }  
  
    public void recalculate(Cell cell) { ... }  
    ...  
}  
  
public class TestingSpreadsheet extends Spreadsheet {  
  
    public void recalculate(Cell cell) {  
        ...  
    }  
}
```

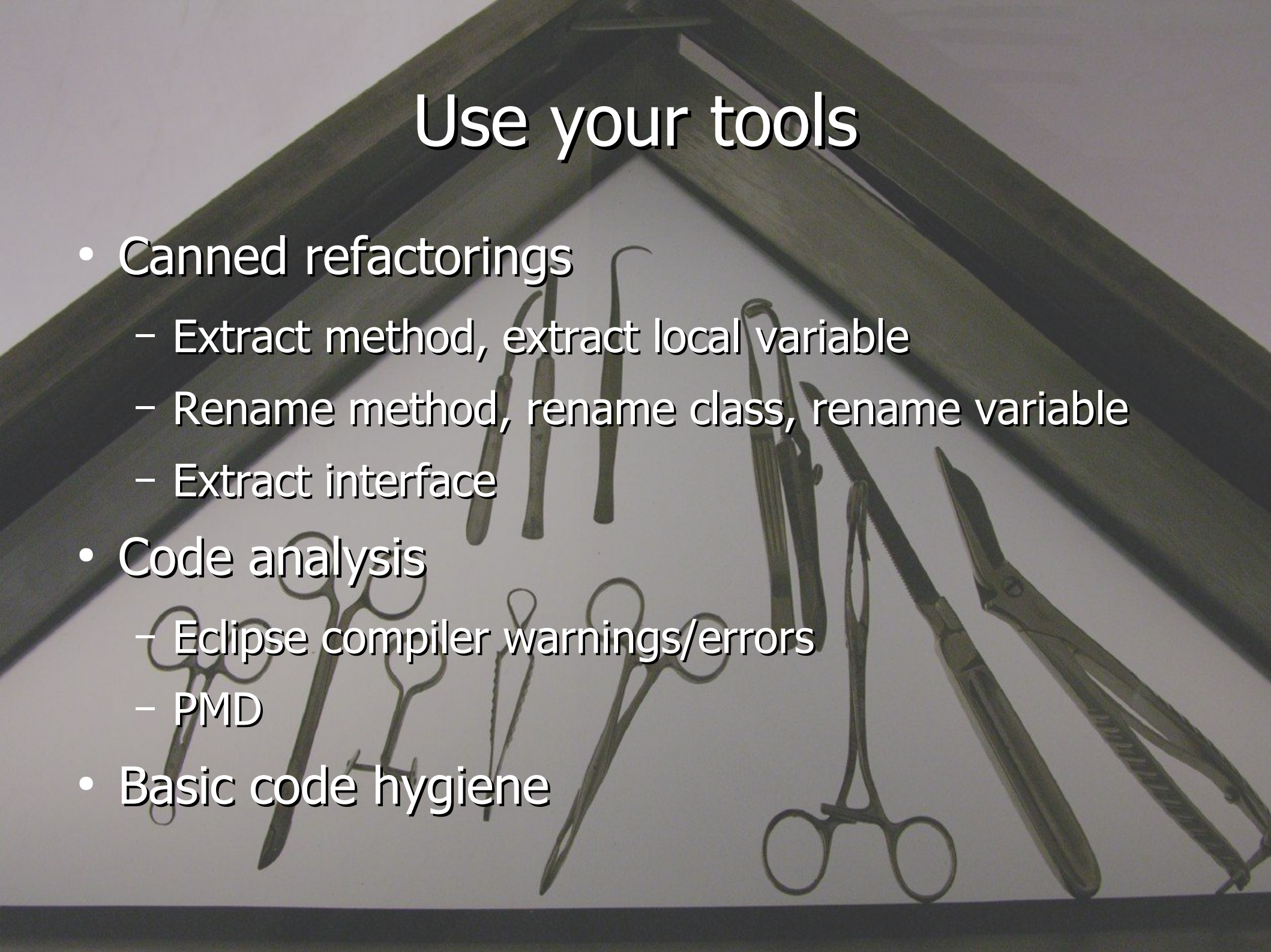
(based on *Working Effectively with Legacy Code*, p. 42/43)

Reveal intentions

A photograph of three men in a workshop or office environment. They are gathered around a laptop, looking intently at the screen. The man in the center, wearing a yellow shirt and glasses, is pointing at the screen. The man on the left, wearing a dark blue shirt and glasses, has his hand to his chin in a thoughtful pose. The man on the right, wearing a colorful patterned shirt, is also looking at the screen. In the background, there are wooden lockers and another person in a white shirt.

Write code for who comes after you,
not for the computer

Use your tools



- Canned refactorings
 - Extract method, extract local variable
 - Rename method, rename class, rename variable
 - Extract interface
- Code analysis
 - Eclipse compiler warnings/errors
 - PMD
- Basic code hygiene

One smell at a time

Code smell
*warning sign about
possible problem*

God class
Long method
Duplicated code
Intention hiding names

Focus!

Refactoring backlog



Pairs & peers



A large excavator is shown in the process of demolishing a multi-story brick building. The excavator's arm is extended, and it appears to be breaking down the structure. In the foreground, there is a massive pile of rubble, including bricks, wood, and other debris. The background shows the remaining parts of the brick building, with several windows visible. The sky is overcast.

Sometimes
it gets worse

before it
gets better

- Defactoring
- Temporarily use not-so-good practices

Refactor as you go

- Exploratory Refactoring
- Clean what you're working on
 - Always leave it in a better state than when you started
 - Add Tests
- Persevere!

CoffeeMUD

- MOB = any creature in the world
- StdMOB
 - Smells: god class, large class, long methods
 - Collaborators: Ability, Behavior, Item, Faction, Deity
 - Interfaces: MsgListener, Environmental

```
public boolean okMessage(Environmental myHost, CMMsg msg) {
    try {
        checkNotDeity(msg);

        checkCharacterStats(msg);

        checkEffects(msg);
        checkInventories(msg);
        checkBehaviours(msg);
        checkFactions(msg);

        MOB mob = msg.source();
        checkImmortalityAndLimbs(msg, mob);
        checkConsequencesOfRange(msg, mob);

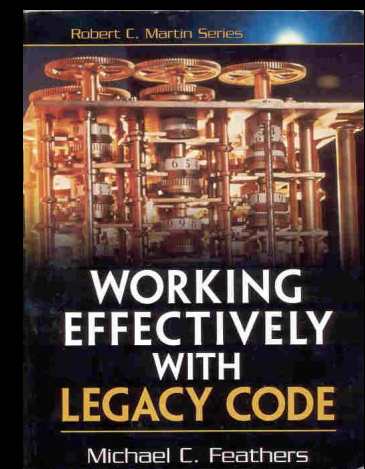
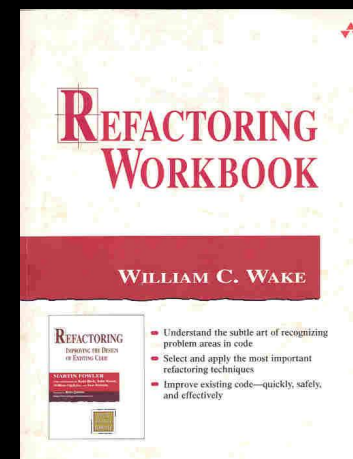
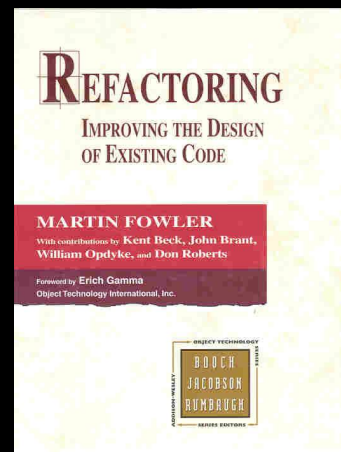
        if (targetHasEffect(msg) && (msg.amITarget(this))) {
            checkIAMPresentInWorld();
            performCombat(msg, mob);

            if ((rangeToTarget() >= 0) && (!isInCombat()))
                setAtRange(-1);

            checkCanDoActionToTarget(msg, mob);
        }
        invokeDepressionAroundMyDeath(msg);
        ...
    }
}
```

References

- Martin Fowler, *Refactoring*
- William C. Wake, *Refactoring Workbook*
- Micheal Feathers, *Working Effectively with Legacy Code*



Copyright

This work is licensed under a Creative Commons Attribution-Share Alike 3.0 Netherlands license - <http://creativecommons.org/licenses/by-sa/3.0/nl/>

Selbsthass © Schnurrbart (CC Attribution-Share alike)
<http://flickr.com/photos/schnurrbart/49048321/>



light at the end of the tunnel © BotheredByBees (CC Attribution)
<http://flickr.com/photos/botheredbybees/1465128757>



house moving © Karen Apricot (CC Attribution-Share Alike)
<http://flickr.com/photos/karenapricot/402072480/>



Brown © trekkyandy (CC Attribution-Share alike)
<http://flickr.com/photos/trekkyandy/1236266577>



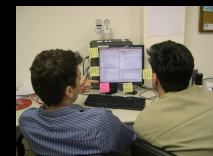
Them tools © Jurek D. (CC Attribution)
http://flickr.com/photos/jurek_durczak/311837936/



Smelly Fish © deejaymarlon (CC Attribution)
<http://flickr.com/photos/deejaymarlon/428580031/>



Bernardo e Elmar. © Improve It (CC Attribution-Share Alike)
<http://flickr.com/photos/improveit/1681875499/>



Look what I have done! © Elsie esq. (CC Attribution)
<http://flickr.com/photos/elsie/485356846/>

