SZ

HOW TO WRITE MAINTAINABLE SOFTWARE

A crash course in code quality and software maintainability

Sieuwert van Otterloo otterloo@gmail.com
06-10509674 @entreprenl

About Sieuwert van Otterloo

SZ

Current activities:

- Startup enthusiast (investor, journalist)
- Software expert (court-registered IT expert, scrum coach, software auditor)
- Strategy consultant (McKinsey & Company, Eden McCallum)

Claims to fame:

- Has assessed 100+ IT systems (election software, railway software, insurance and banking systems and ATM software)
- Has interviewed more than 100 startup founders to give better advice. The goal is to interview founders from all NL-based startups
- First round investor in several startups (SOWISO, Buzzoek, Kidswatcher, Arete Caerus, Plot Projects, SellanApp)

STARTUPJUNCTURE ALL DUTCH STARTUP NEWS

Computer issues delay flights in Los Angeles

SZ

May 1, 2014 8:51 AM

LOS ANGELES (AP) — Flights to and from airports in the Los Angeles area were grounded for more than an hour Wednesday due to a computer failure at an air traffic control facility in the region, the Federal Aviation Administration said.

The problems rippled nationwide. Dozens of planes heading into the region were diverted elsewhere, and flights scheduled to take off to the Los Angeles area were held on the ground across the country.

[…]

A notice posted on the FAA website said planes were not allowed to depart Los Angeles because of a failure within the agency's En Route Automation Modernization system, also known as ERAM. […]

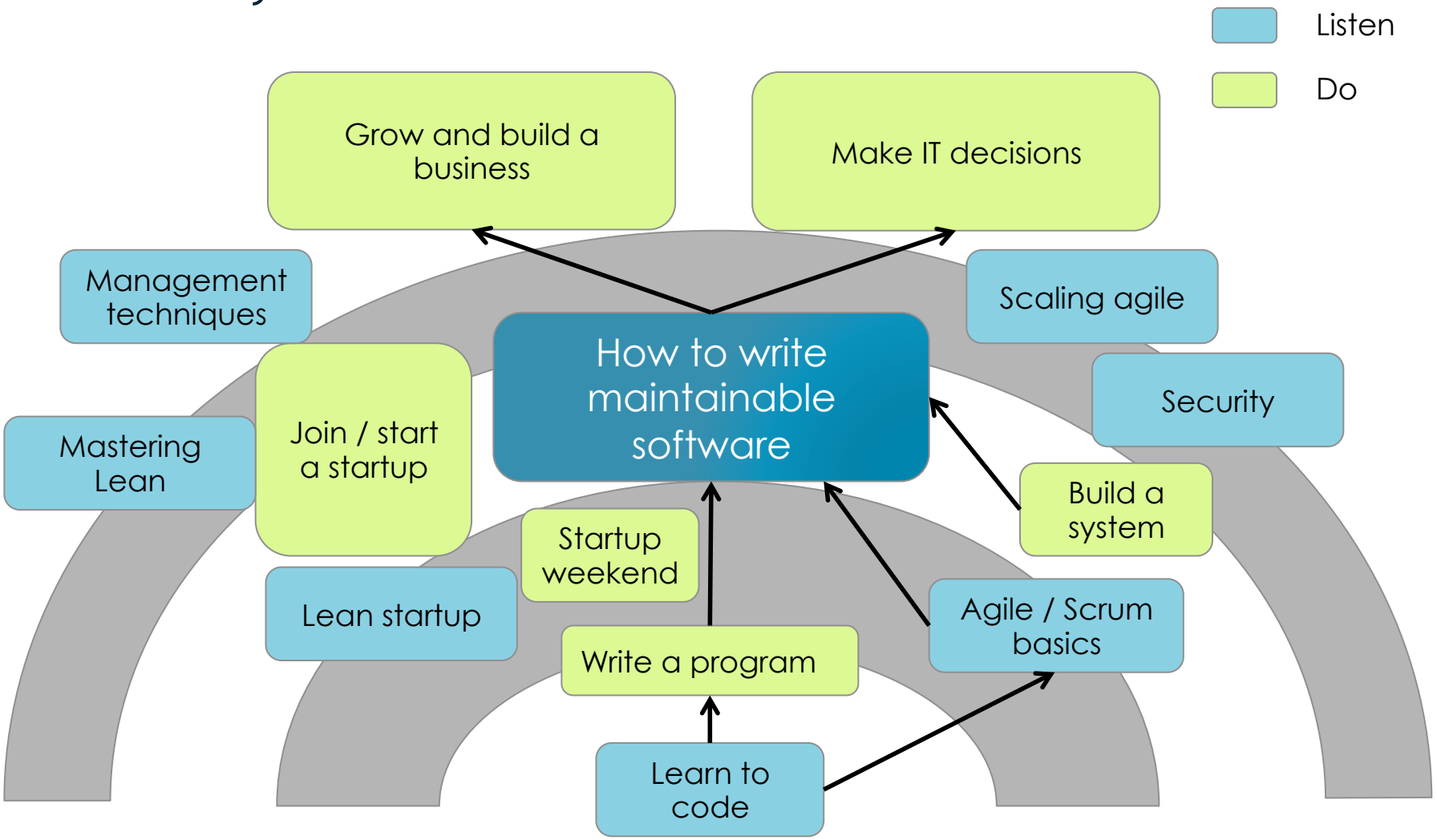
The ERAM system is critical to the FAA's plans to transition from a radar-based air traffic control system to satellite-based navigation, but its rollout is **years behind schedule and hundreds of millions of dollars over budget**.

ERAM is replacing another computer system that was so old that **most of the technicians who understood its unique computer language have retired**.

http://news.yahoo.com/computer-issues-delay-flights-los-angeles-234300027.html

Ideal position of this course in your personal learning plan

SZ



Agenda

SZ

- **The motivation: Why software maintainability is important**
- The destination: What does maintainable code look like
- The road: How to put quality into the software development process
- The roadblocks: How to deal with quality assessments

Why software is important

SZ

Marc Andreessen
Founder of Netscape,
Venture capitalist

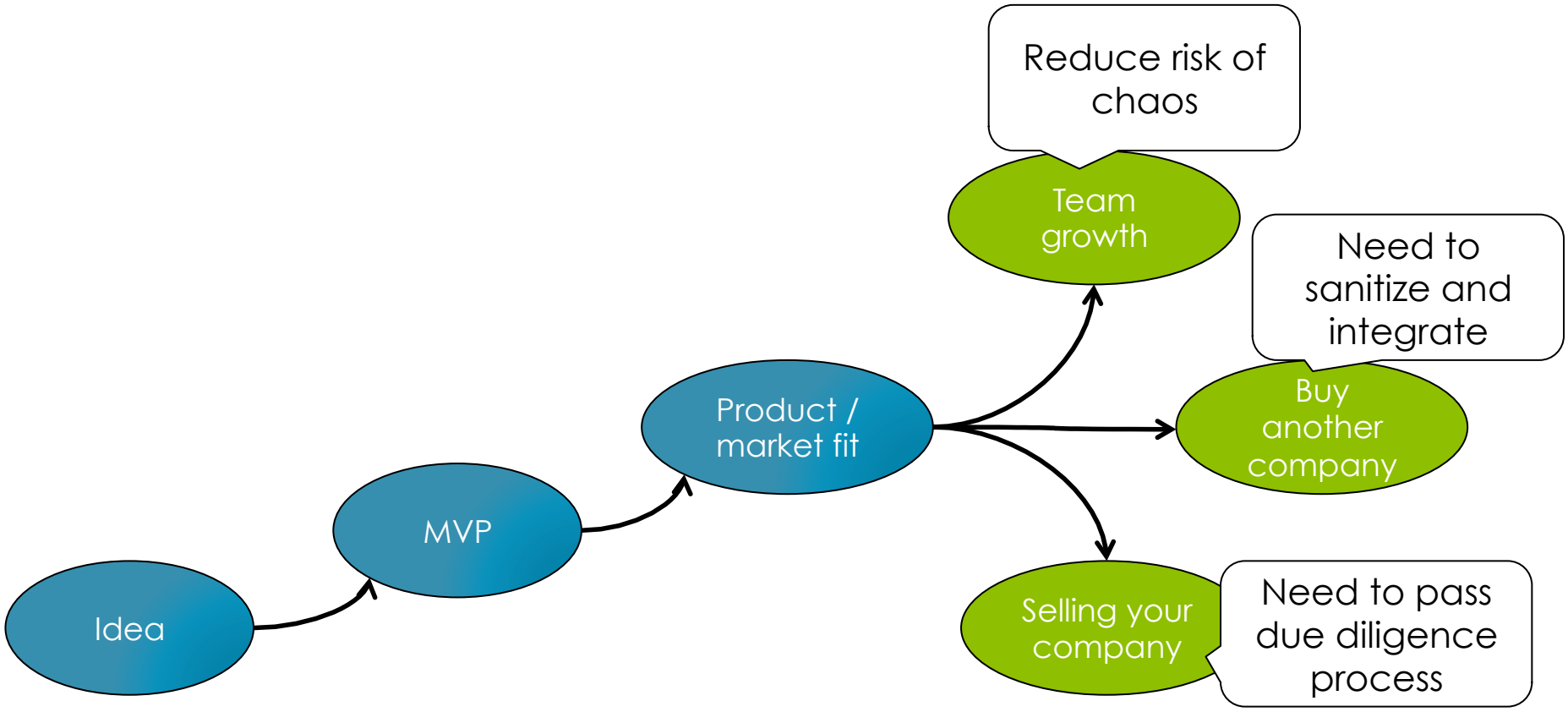
Software is eating the world:

- *Today, the world's largest bookseller, Amazon, is a software company*
- *Today's largest video service by number of subscribers is a software company: Netflix.*
- *Today's dominant music companies are software companies, too: Apple's iTunes, Spotify and Pandora.*
- *Today's fastest growing entertainment companies are videogame makers*
- *The best new movie production company in many decades, Pixar, was a software company.*
- *Today's largest direct marketing platform is a software company—Google.*
- *Today's fastest growing telecom company is Skype*
- *LinkedIn is today's fastest growing recruiting company..*

Source: Wall street journal, August 20 2011

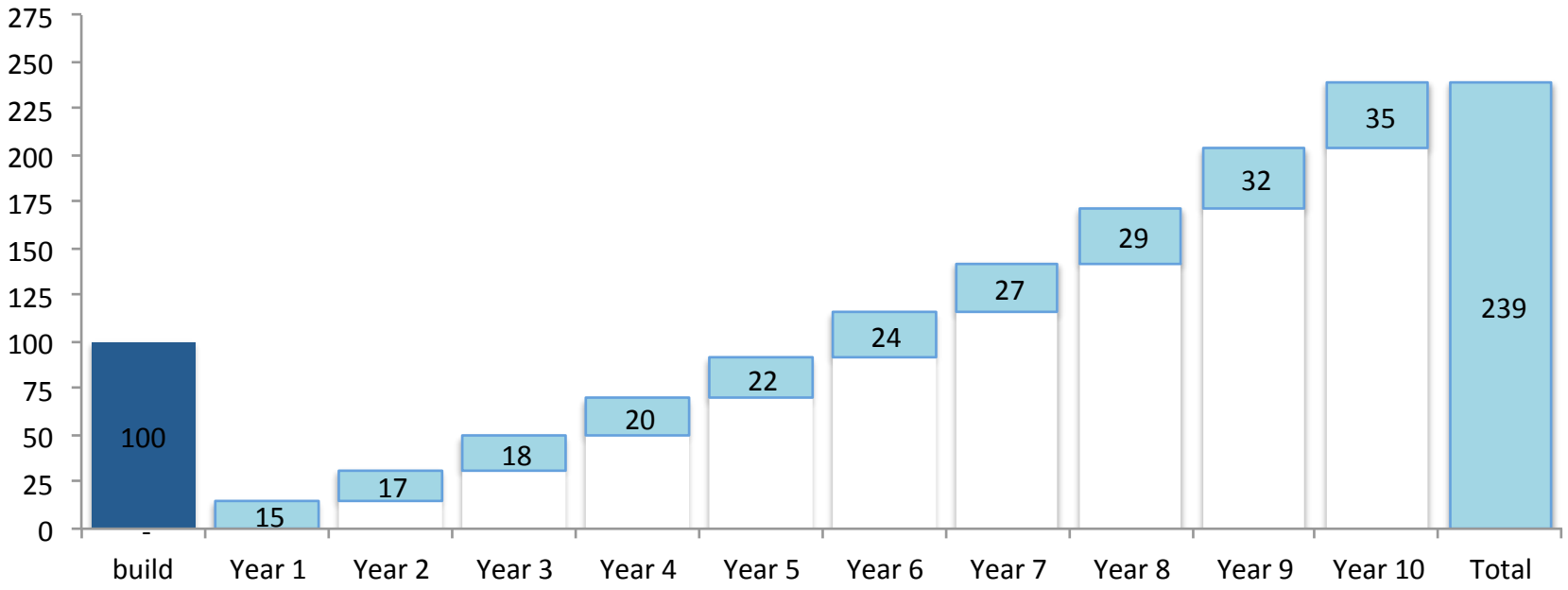
Software maintainability is important for scaling startups

SZ



The classic view on software costs

SZ



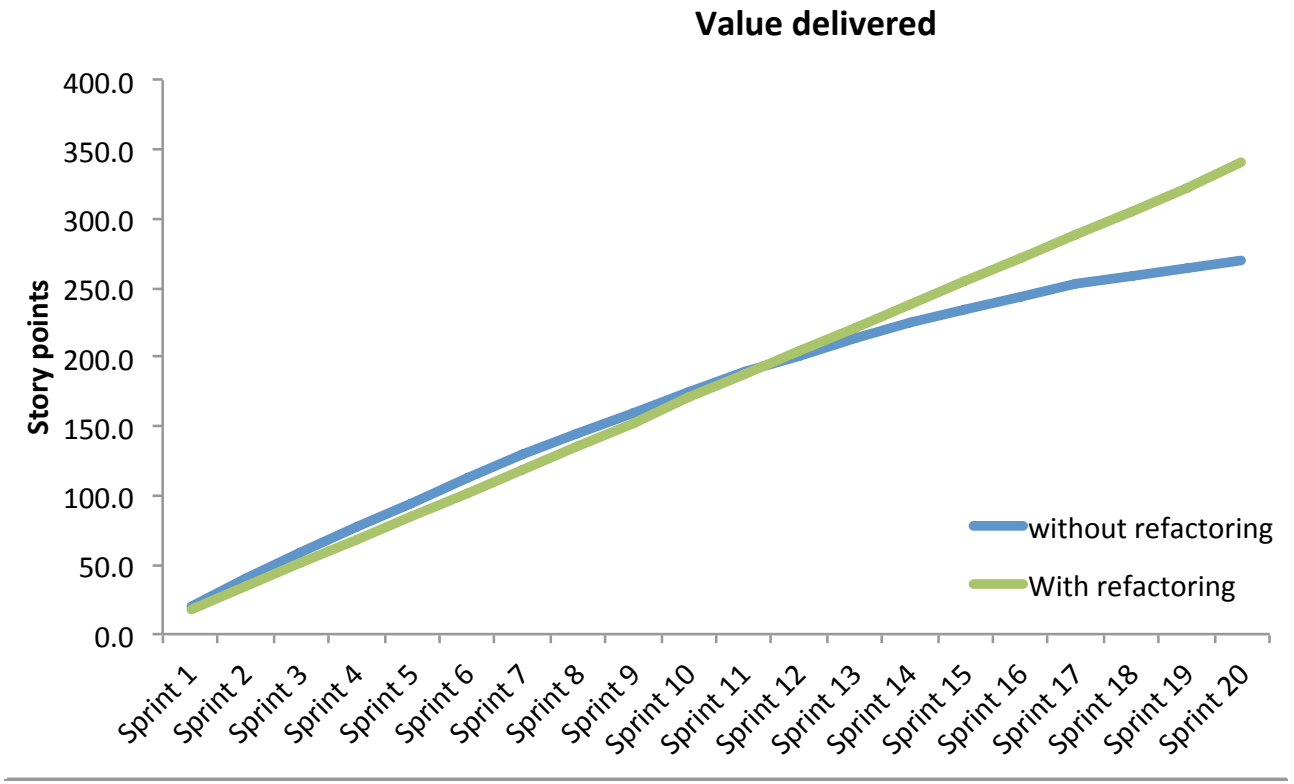
Conservative example:

- The system needs 15% maintenance per year
- The system grows 10% per year
- System lasts 10 years

Result: maintenance costs are 140% higher than development cost

Agile view on software quality: technical debt

SZ



- Technical debt are issues that the customer cannot see, but the developers can
- You can either clean it up (refactor) or work around it
- If you do not clean up, you pay interest

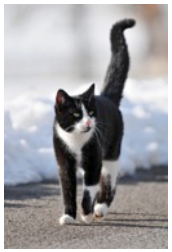
Agenda

SZ

- The motivation: Why SW maintainability is important
- **The destination: What does maintainable code look like**
- The road: How to put quality into the software development process
- The roadblocks: How to deal with quality assessments

Volume

SZ

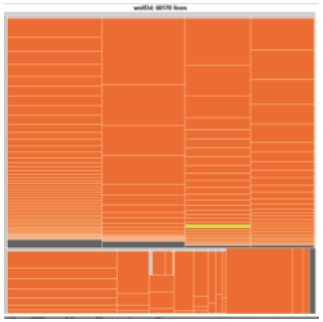


Very small	Nice and small	Hard to handle	Impossible
< 10.000 lines of code	< 100.000 lines of code	Less than 500.000 lines of code	>500.000 lines of code

Understanding the scale

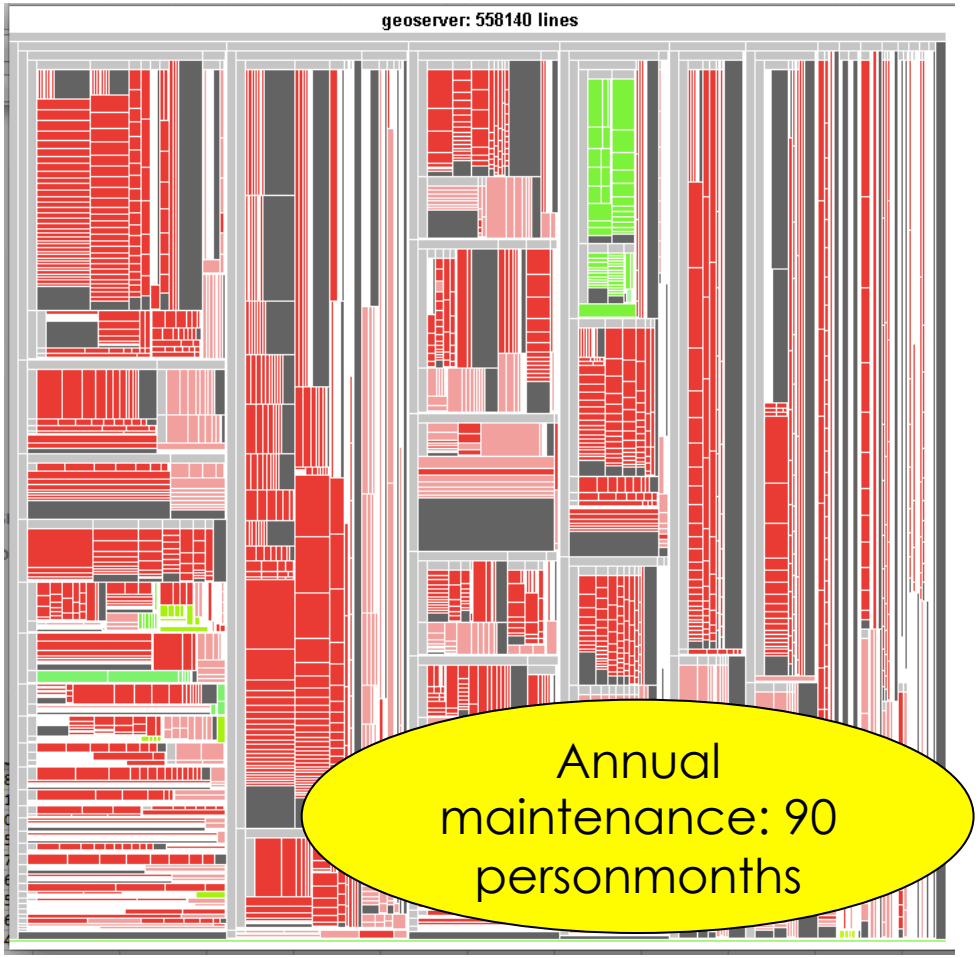
SZ

Small system (60.000 LoC)



Annual maintenance: 9 personmonths

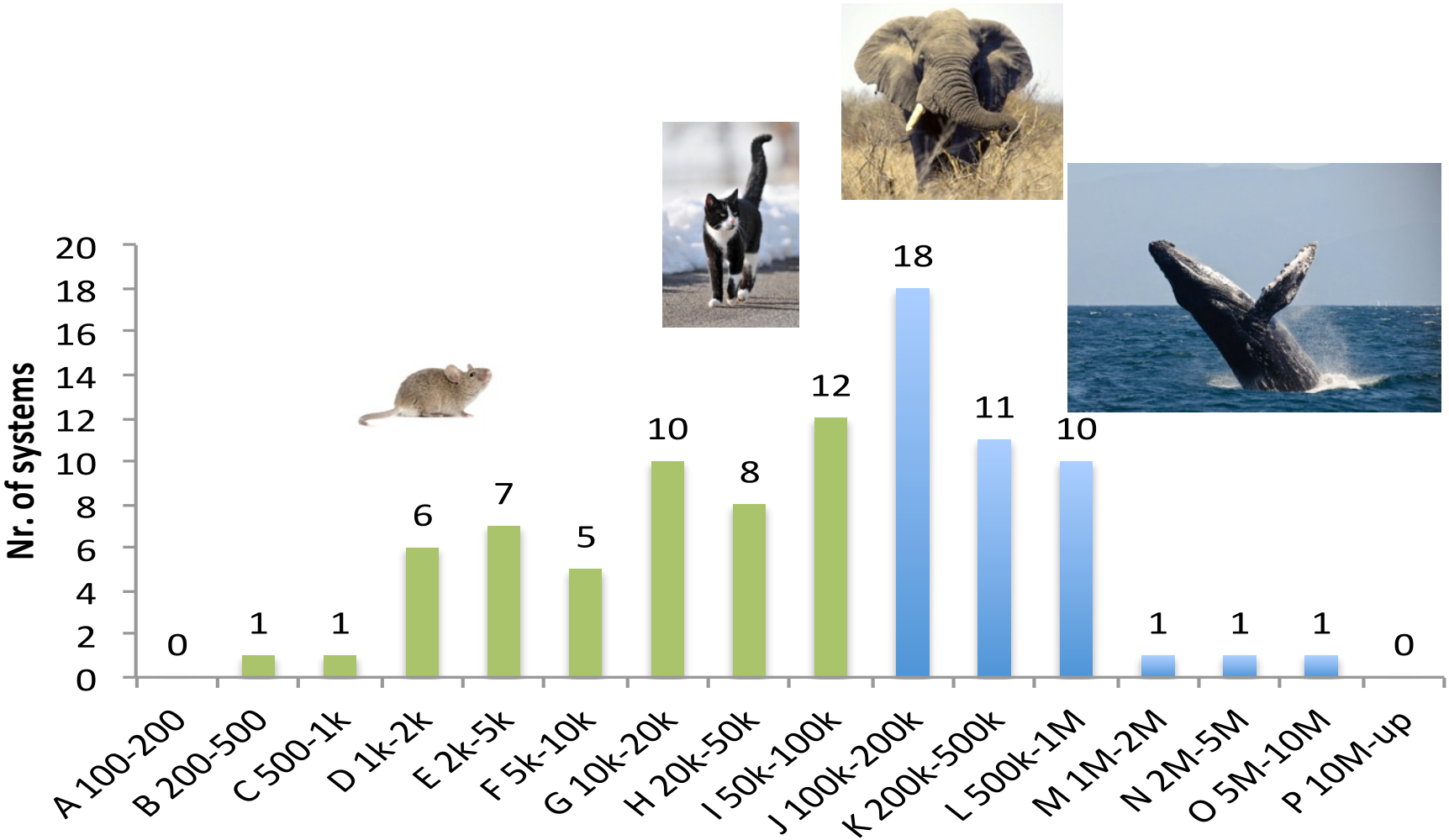
Huge system (600.000 LoC)



Annual maintenance: 90 personmonths

System volume

SZ



Example of a very small program

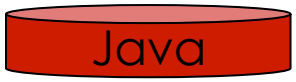
SZ

```
#!/bin/perl -sp0777i<X+d*lMLa^*lN%0]dsXx++lMlN/dsM0<j]dsj
$/=unpack('H*',$_);$_=`echo 16dio\U$k"SK$/SM$n\EsN0p[lN*1
lK[d2%Sa2/d0$^Ixp"|dc`;s/\W//g;$_=pack('H*',/((..)*)$/)
```

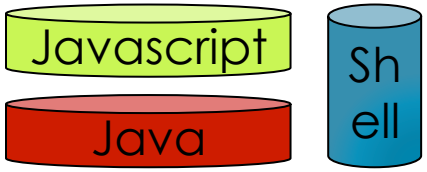
Use of technologies

SZ

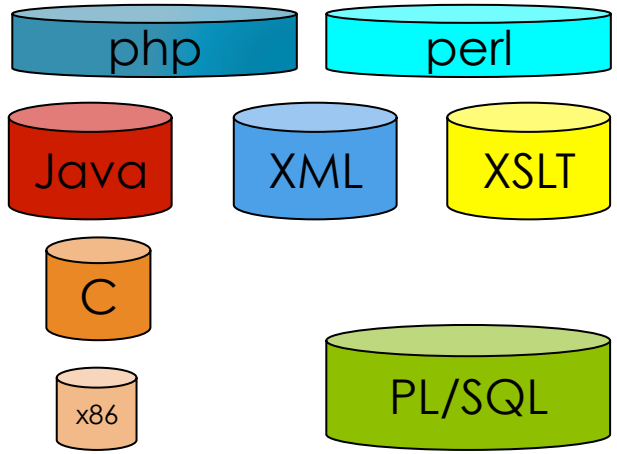
Simple stack



Complicated stack

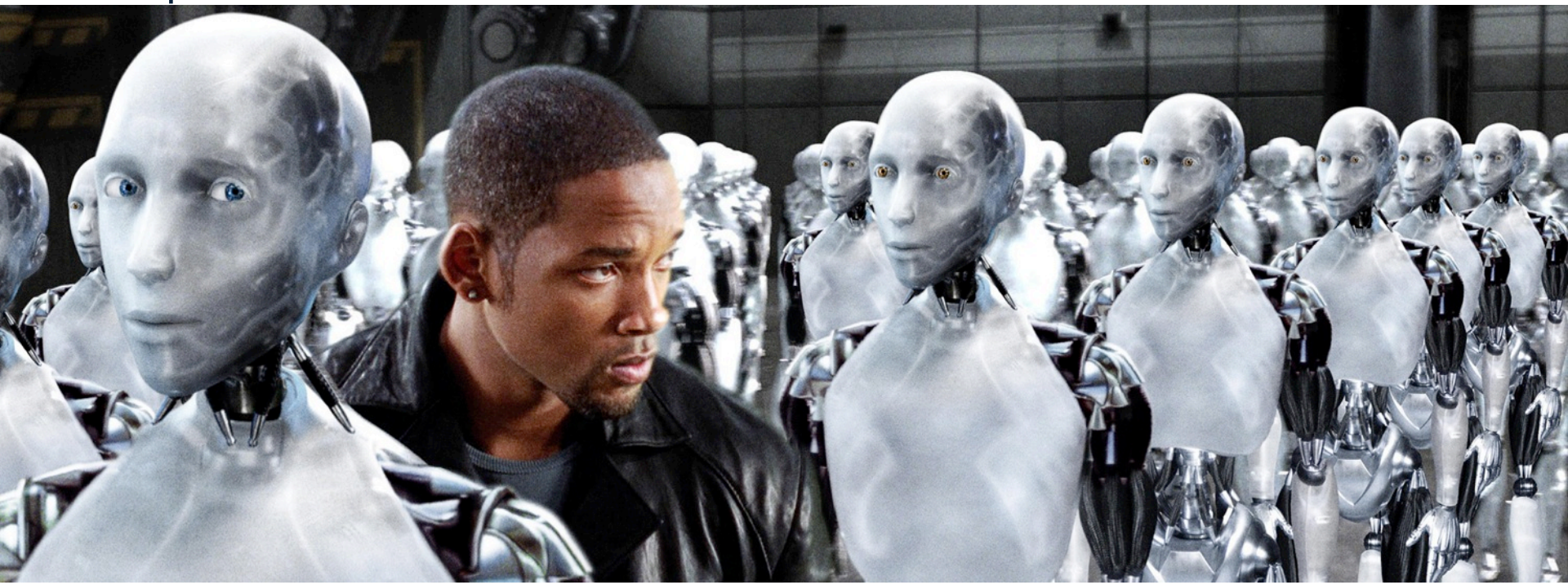


'Legacy' stack



Duplication

SZ



Found 185 duplicate lines in the following files:
Between lines 29 and 235 in /java/jabref-2.9.2/src/java/net/sf/jabref/export/layout/format/FormatChars.java
Between lines 31 and 239 in /java/jabref-2.9.2/src/java/net/sf/jabref/oo/OOPreFormatter.java
Found 194 duplicate lines in the following files:
Between lines 130 and 397 in /java/jose-144-source/java/de/jose/util/Metaphone2.java
Between lines 129 and 396 in /java/jose-144-source/java/de/jose/util/Metaphone.java

Complexity: decision points per method

SZ

Best: less than 7 decision points per method (128 paths)

Mediocre: less than 10 (1024 paths)

This code: 36 decision points (68,719,476,736 paths)

```
public boolean equals(Object obj) {
    if (obj instanceof ComparableAppearance) {
      Appearance appearance2 = ((ComparableAppearance)obj).appearance;
.....
                if (!color1.equals(color2)) {
                  return false;
                } else if (material1.getShininess() != material2.getShininess()) {
                  return false;
                } else if (material1.getClass() != material2.getClass()) {
                  return false;
                } else if (material1.getClass() == OBJMaterial.class) {
                  OBJMaterial objMaterial1 = (OBJMaterial)material1;
                  OBJMaterial objMaterial2 = (OBJMaterial)material2;
                  if (objMaterial1.isOpticalDensitySet() ^ objMaterial2.isOpticalDensitySet()) {
                    return false;
                  } else if (objMaterial1.isOpticalDensitySet() && objMaterial2.isOpticalDensitySet()
                            && objMaterial1.getOpticalDensity() != objMaterial2.getOpticalDensity())
                  {

                    return false;
                  } else if (objMaterial1.isIlluminationModelSet() ^
objMaterial2.isIlluminationModelSet()) {
                    return false;
                  } else if (objMaterial1.isIlluminationModelSet() &&
objMaterial2.isIlluminationModelSet()
                            && objMaterial1.getIlluminationModel() !=
objMaterial2.getIlluminationModel()) {
                    return false;
                  } else if (objMaterial1.isSharpnessSet() ^ objMaterial2.isSharpnessSet()) {
                    return false;
                  } else if (objMaterial1.isSharpnessSet() && objMaterial2.isSharpnessSet()
                            && objMaterial1.getSharpness() != objMaterial2.getSharpness()) {
                    return false;
                  }
                }
              }
            }
          }
        }
```

Source: SweetHome 3D, fileOBJWriter.java

Other important aspects

SZ

- Missing exception handling
- TODO comments
- Long ‘do-it-all’ files
- Memory actions and leaks
- Safe use of user strings
- Complex queries
- Code copyrighted by others

Typical quality model (SIG)

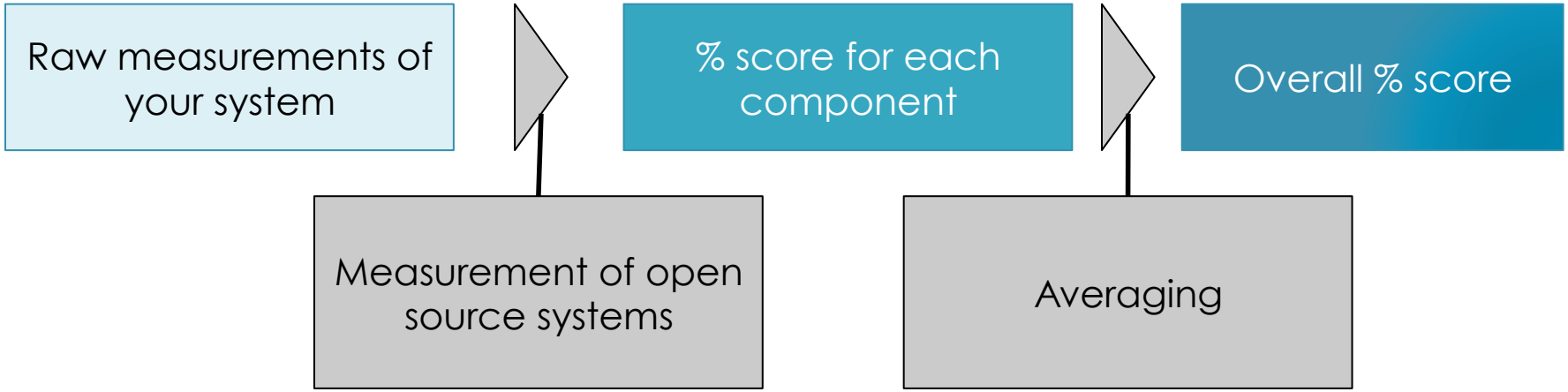
SZ

Maintainability sub-characteristic / Product property	Volume	Duplication	Unit size	Unit complexity	Unit interfacing	Module coupling	Component balance	Component independence
Analyzability	X	X	X				X	
Modifiability		X		X		X		
Testability	X			X				X
Modularity						X	X	X
Reusability			X		X			

SIG/TÜViT Evaluation Criteria Trusted Product Maintainability • Version 5.1

Benchmarking quality

SZ



"120 violations of type X"

"Aspect X is better than 55% of systems"

"Maintainability is better than 60% of systems"

Agenda

SZ

- The motivation: Why SW maintainability is important
- The destination: What does maintainable code look like
- **The road: How to put quality into the software development process**
- The roadblocks: How to deal with quality assessments

Joint code ownership

SZ

- Everyone in the team should feel comfortable explaining each line of code
- All founders should be interested in the code on which the company runs

Create daily feedback

SZ

Structure

- Integrate and test a working system at least every two weeks
- Agree on basic code quality standards

Tools

- Create a fully automated daily build process
- Use automated tools in the build process (checkstyle, FxCop, Simian, PMD, Sonar)
- Configure automated checks in the IDE (Visual studio)

Mindset

- Monitor issues daily as a lead developer
- Address root causes of issues in retrospectives:
  - Training needs for new and current developers
  - Important refactoring actions
  - Adjustments to quality standards

Five steps for a quality program

SZ

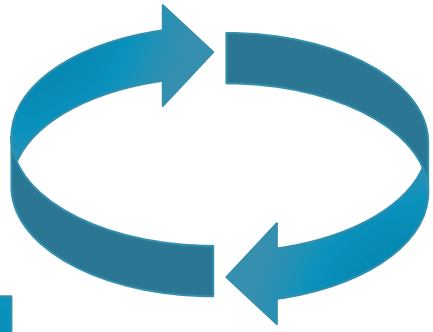
1. Make strategy and technology choices

2. Set quality standards and architecture rules

3. Start measuring

4. Create good new code

5. Clean up old code



Agenda

SZ

- The motivation: Why SW maintainability is important
- The destination: What does maintainable code look like
- The road: How to put quality into the software development process
- **The roadblocks: How to deal with quality assessments**

Top 4 situations when assessments / audits happen

SZ

Product focus

Company focus

1. A large company buys your service

3. Someone invests in your company

2. You are supplier in a delayed/troubled project

4. Someone buys your company

Software Risk Assessment process

SZ

How not to deal with an assessment

SZ

Client | **Assessor** | **Supplier**

Develop a system as fast as possible at minimal cost

OK, here it is

Can you audit the system?

What quality standards did you demand?

We asked nothing special, but we expect a fit for use system conforming to industry best practices

What quality standards did you use?

None, we focused on cost and speed

Let's report a lot of findings to show that we worked really hard

A better way to deal with assessments

SZ

Client | **Assessor** | **Supplier**

Develop a system as fast as possible at minimal cost

Here is our own standard, is that good enough for you?

What quality standards did you use?

We agreed on this standard. We checked to code and it complies. Let us know if you find any issues

We worked really hard and have these findings

Well done! We do not see major risks, but if needed we have a quality process and can fix these in the next release.

The quality is what has been agreed, and will be even better in the next release

How to deal with due diligence

SZ

1. Keep it short by starting late: **Do not start the assessment** before the other deal details are sorted out
2. Ensure the goal is limited: For instance to determine whether the software has issues that **cannot be fixed** and cause **major** risks
3. Ensure **involvement**: Auditors should listen to your side, share and discuss findings before reporting any issues.

Conclusions

SZ

- **Software quality is important** for any growing or grown company
- **You can achieve quality** with the right agile development process

Thank you!

SZ

- These slides will be made available on www.softwarezaken.nl
- If you have any questions, contact me
- I can be hired for additional workshops, coaching and consulting on any of the topics below:

Call or mail me:
otterloo@gmail.com
+31 6 1050 9674

Lean startup	Lean startup for corporates	Startup search & selection
Starting with agile / scrum	maintainable software	Secure software development
IT management for non-IT	IT strategy	IT contracts