Learning from Mistakes

Andreas Zeller

Fixing the Process

- Any defect escaping into the wild should have been caught by local *quality assurance*
- Besides fixing the defect, we also must fix quality assurance!

Things to do

- Improve your test suite
- Set up assertions
- Improve training
- Improve the software process
- Improve the analysis tools

Things to Measure

- How much damage did the defect do?
- How much effort did it take to fix it?
- What is the risk we are taking in letting such defects go unnoticed?

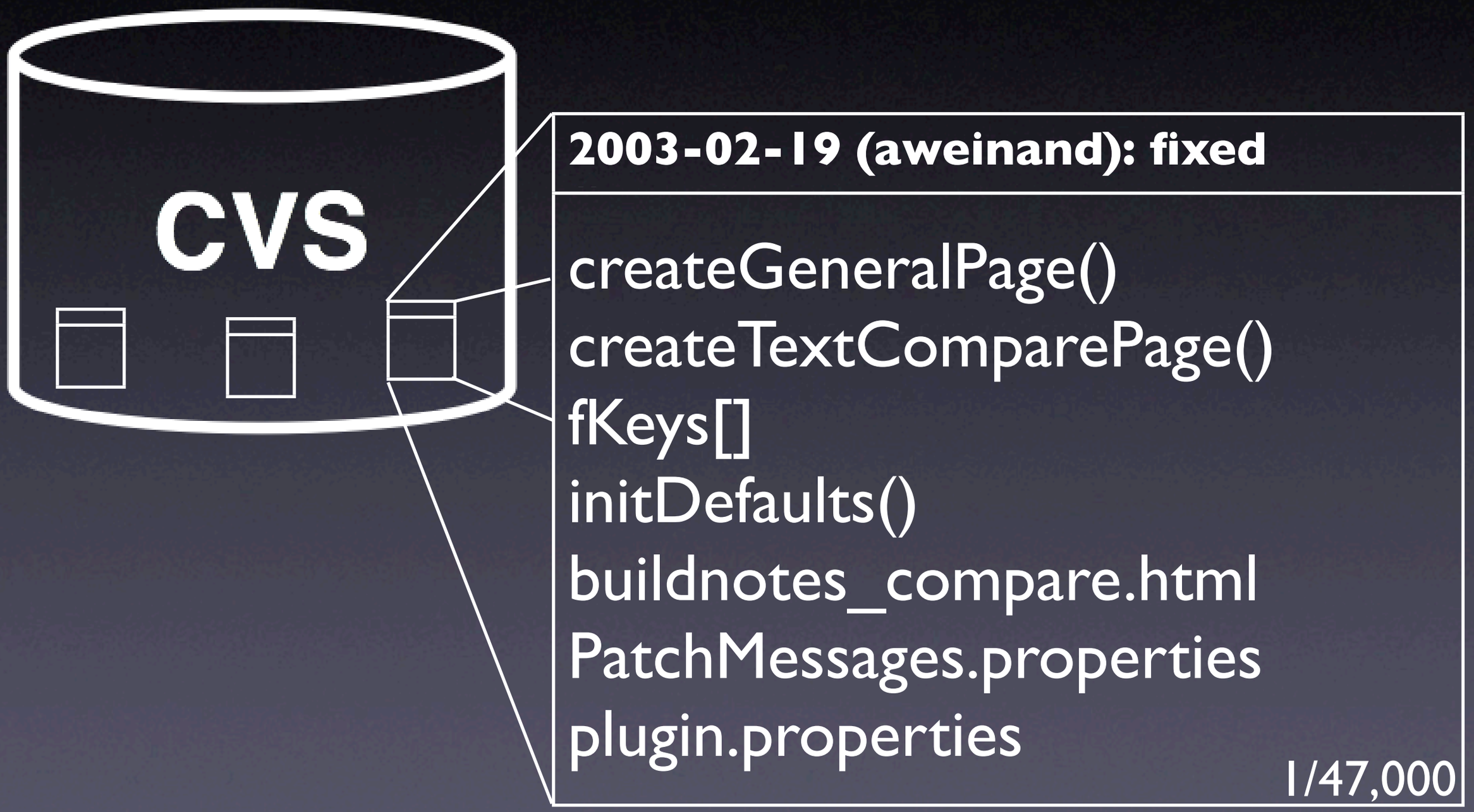
Some Facts

- *In Eclipse and Mozilla, 30–40% of all changes are fixes* (Sliverski et al., 2005)
- *Fixes are 2–3 times smaller than other changes* (Mockus + Votta, 2000)
- *4% of all one-line changes introduce new errors* (Purushothaman + Perry, 2004)

More Facts

- *A module that is one year older has 30% less errors* (Graves et al., 2000)
- *New code is 2.5 times as defect-prone as old code* (Ostrand + Weyuker, 2002)

Learning from History



Requirements

- Well-kept version and bug databases
- Link between changes and problems
- Willingness to change
- Policy on how to handle sensitive data

Space Shuttle Software

Problem Tracking

- When was the error discovered? How? Who? What flight?
- How was the error introduced? Why wasn't it caught?
- How was the error corrected? Are there similar errors?
- What can we learn from previous errors?

The Process

- Software error = *an error in the process*
- Planning the software carefully in advance
- Reducing risk at all stages
- Keeping record of all activities
- “Not even rocket science” – just standard practice in engineering