



# The English Breakfast Network

## KDE quality checking framework

Adriaan de Groot

October 7, 2006



# Who is this Guy?



## Professional

- KDE contributor since 2000
- KDE Quality Team Lead since 2005
- Researcher in Software Quality since 2004

## Private

- Father of two
- Recumbent cyclist



## APIDOX Problems

- They were incomplete,
- they were generated infrequently,
- they contained errors.

## ... solutions

- Build them more often,
- keep the logs and examine them.



### User Documentation Sanitizer

i18n-doc team (Phil Rodrigues, Frerich Raabe) and the documentation guidelines.

### Coding Standards Checking

Coding guidelines from Ben Meyer, Frans Englisch with checking tools. Now maintained by Allen Winter.

### Usability Testing

The *quantifiable* parts of the HIG could be tested by an automatic tool. Candidate tool is FrogLogic's Squish.



## Detect and report

- The purpose of the EBN (toolset) is to *detect* and *report* defects in the KDE codebase.
- The purpose of the EBN is to say something about the *quality* of the KDE codebase.
- The purpose of the EBN is to indicate where developer activity would pay off most.



### Exterminate; exterminate

Detect and report defects in order ...

- to give developers some direction,
- to show low-hanging fruit,
- to give developers a good starting point,
- to show objective improvement.

### Scope creep

- Fun activity pictures,
- “one-stop shop” for state information.



## Ad-hoc sweetness

- Shell scripts drive the process,
- Doxygen / Perl scripts / Squish do checks,
- Perl scripts post-process logfiles,
- Python scripts do visualisation.

Result *counts* get dumped into a database.



## Defect Views

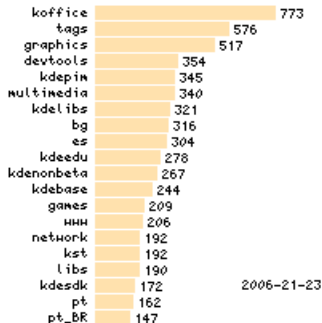
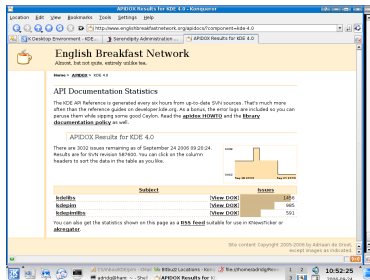
- Counts per module,
- counts per subdirectory,
- details per subdirectory.

## What's Missing

- *All* details per subdirectory,
- historical view,
- sense of “software unit.”



# Implementation (3)



2006-21-23



# What would developers do?



## Use the EBN

- To steer own efforts,
- to monitor other efforts,
- to learn about guidelines.

## Ignore the EBN

Because ...

- not the right information,
- presentation sucks,
- defect measures irrelevant.



## Tools

Bring in parsing and semantic tools. Do static checking. Dashboard development and perceptual research.

## SQO-OSS Project

Using static checking, semantic analysis, and change-based quality metrics, do *project* quality checking. Targeting many projects, with “soft indicators” taken into account.



## Re-architecting

- Improved granularity in the DB,
- searchable results,
- collating results across tools,
- cmake-style dashboard,
- RSS subscriptions for applications.



Now what?



Questions ?