Universität Berlin

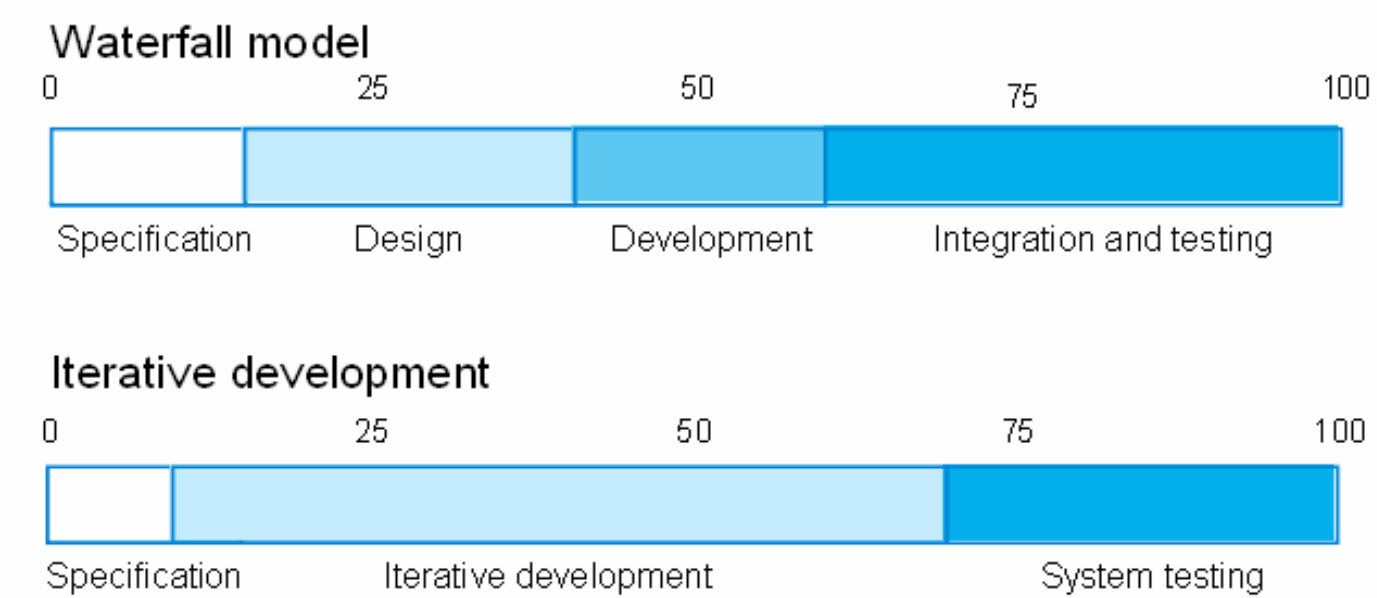
Doktorandenworkshop 2004

Human Error and Defect Prevention

Sebastian Jekutsch
Freie Universität Berlin
Institut für Informatik
Arbeitsgruppe Software Engineering

Costs in Software Development

- Software development:
  60% development costs, 40% testing costs



Usual Software Quality Assurance

- Testing = Causing the software to fail
- Debugging = Searching and removing defects

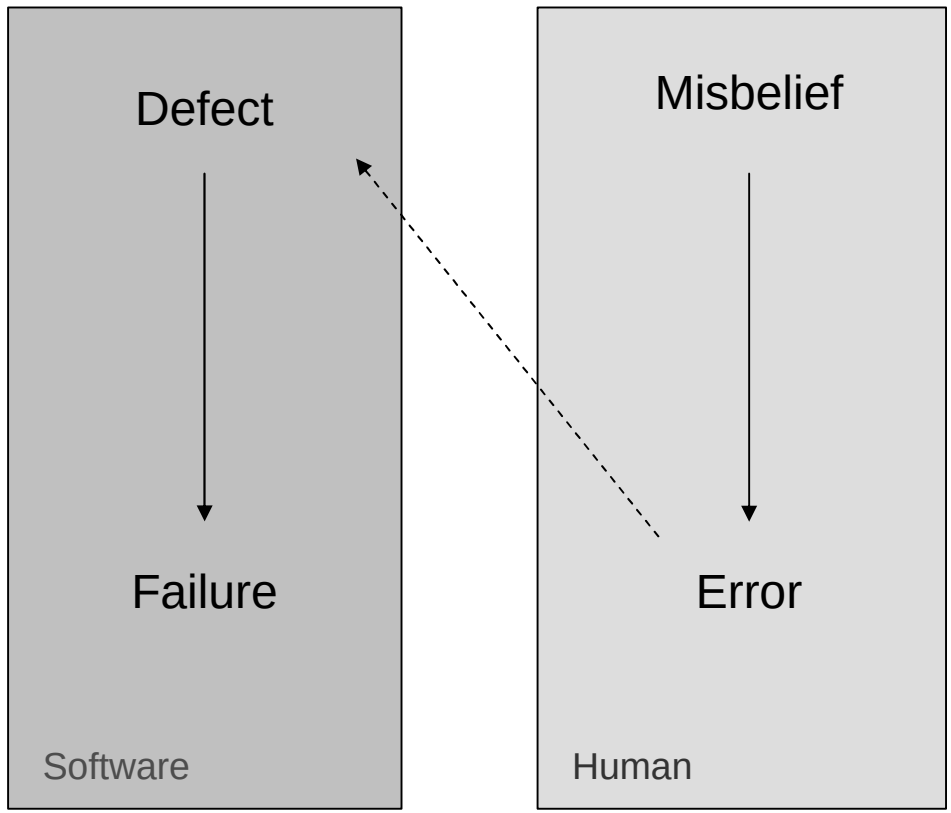
- So...
  First we insert defects,
  then we are searching for them,
  then we are removing them.

- And...
  We are fighting symptoms!

Interaction, Terminology

Cause / State

Effect / Event



Research questions

- Why are we *creating* defects?
- What are the reasons for making programming/design errors?
- What are the kinds of misbeliefs in software development?

- How can we detect errors or even misbeliefs?
- How can we prevent inserting defects?

Research strategy: Triangulation

1. Psychological work on human error
2. Classification of defects and errors
3. Capturing the micro-process of software development

1. Psychological work on human error

- Relating psychology…
  - Psychologist's theories on human error
  - Typical errors in reasoning and planning
- …to software engineering
  - investigating typical *programming* errors
  - examining cognitive biases in software development
- To explain programming errors with general results on studying human errors.

2. Classification of defects and errors

- Learning about errors is learning about defects
- No general defect classification has been presented so far
- ... nor error classifications

- Defect classification $\Rightarrow$ error classification

- Software archeology for mining defects

3. Micro-process of software development (1)

Observing programmer while programming/making errors:

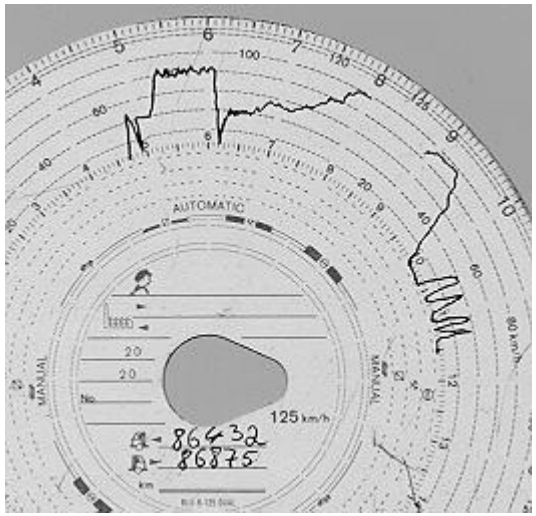
i. Capturing the programming events like
   - changing code parts
   - browsing code
   - pausing, etc.

ii. Grouping on episodes like
   - trial-and-error-cycles
   - copy-paste-change habits
   - interrupted work

iii. Anti-Patterns of programming episodes
   - changing small part of code very often
   - being interrupted and not resuming work well
   - doing trial-and-error for quite a while

3. Micro-process of software development (2)

Other opportunities using micro-processes



- When X was an defect insertion, ~X may be also
- Tracking evolution of code copies
- “Macro-fying” work episodes
- Suggest places to look at because of past browsing sessions
- Re-examining past coding sessions
- Evaluating new metrics, learning

- Logging programmer activities in Eclipse

Looking for help

- This is mainly empirical work
- Therefore, I need to observe programmers
- In non-trivial projects

- Maybe at inf.fu-berlin?

Thank you!

Sebastian Jekutsch
Room 008
Tel: 030 / 838-75239
http://www.inf.fu-berlin.de/~jekutsch