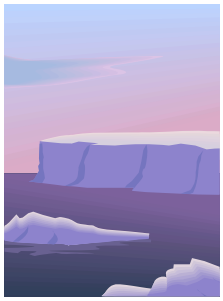
DEFECT-FLOW MODELS

Tutorial and Workshop, Tokyo, 11.12.2009



Presented by
Michael Kläs and Dr. Jens Heidrich

Fraunhofer IESE

Fraunhofer IESE

Welcome – About the Fraunhofer Gesellschaft

Named after

- Joseph von Fraunhofer (1787-1826), a successful researcher, inventor and entrepreneur

Role of the Fraunhofer Gesellschaft

- Germany's leading organization for applied research and technology transfer

Size

- 58 institutes
- Approx. 12.500 employees

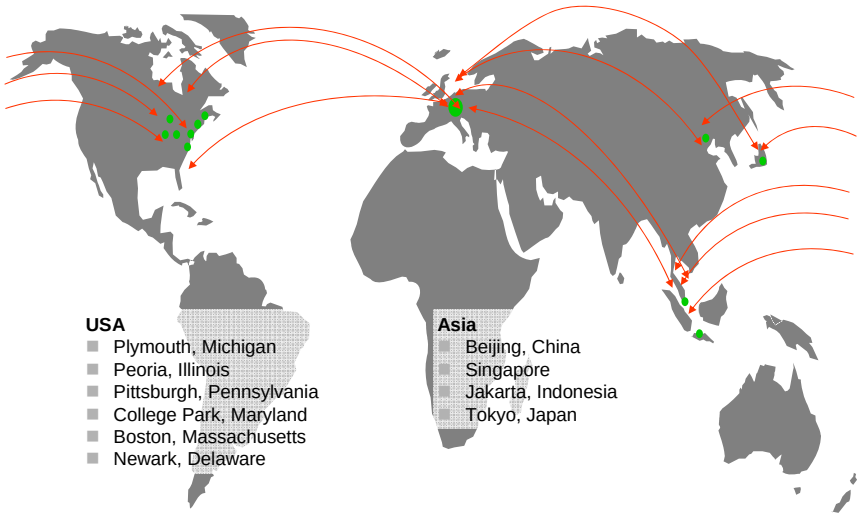
Funding Volume

- about € 1.3 billion
- 1/3 base funding (government)
- 1/3 industrial projects
- 1/3 public sector projects

Fraunhofer Research Units in Germany

- Institutes
- Branches of Institutes, Research Institutions, Working Groups, Branch Labs, and Application Centers

Fraunhofer Locations Worldwide



About Fraunhofer IESE

Principles

- Transferring proven technologies into practice
- Applying empirical methods to evaluate processes and products
- Identifying improvement areas and proposing changes
- Utilizing experience to guide technical and management choices

Measurement Services

- Defining and optimizing quality assurance strategies
- Introducing and optimizing measurement systems
- Establishing and improving estimation capabilities
- Assessing products and processes
- Introducing measurement-based software process improvement
- Training and coaching

About the Presenters (1/2)

Michel Kläs

Researcher at the Processes and Measurement department (PAM) at the Fraunhofer Institute for Experimental Software Engineering (IESE), Kaiserslautern, Germany

Research focus

- Defect prediction & classification
- Software cost estimation
- Goal-oriented measurement
- Empirical software engineering

About the Presenters (2/2)

Dr. Jens Heidrich

Head of the Processes and Measurement department (PAM) at the Fraunhofer Institute for Experimental Software Engineering (IESE), Kaiserslautern, Germany

PAM focus on

- Goal-oriented Measurement
- Project Control Centers
- Domain-specific Quality Models
- Process Management and SPI

Introduction of Tutorial Participants

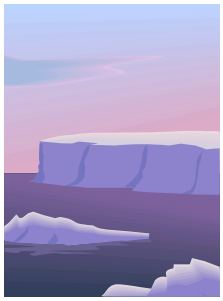
- Name
- Role in your organization
- Prior knowledge and experience
- Expectations regarding the tutorial

Agenda

Tutorial:	**Defect Flow Models**
10:00-10:15	Part 1: Welcome and Introduction
10:15-11:15	Part 2: Defect Classification Foundations
11:15-12:30	Part 3: Defect Flow Model
12:30-13:30	*Lunch break*
Workshop:	**Applying Defect Flow Models**
13:30-14:30	Part 4: Overview on Industrial Case Studies
14:30-16:00	Part 5: Practical Exercises

DEFECT-FLOW MODELS

PART 1: Introduction



Dr. Adam Trendowicz
adam.trendowicz@iese.fraunhofer.de

Martin Kowalczyk
martin.kowalczyk@iese.fraunhofer.de

Michael Kläs
michael.klaes@iese.fraunhofer.de

Ove Armbrust
ove.armbrust@iese.fraunhofer.de

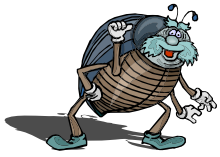
Fraunhofer IESE

Defect measurement

"I can only think of one metric that is worth collecting now and forever: defect count.

Any organization that fails to track and type defects is running at less than its optimal level."

Tom deMarco



Fraunhofer IESE

Why spend effort on defects?

- Defects are major cost drivers in the software industry
- Post-delivery defects in software may endanger complete systems
- Software failures caused by defects may trigger enormous costs
- Software failures caused by defects may result in human casualties
- The later a defect is found, the more expensive it is to correct
- Examples…

Software quality disasters: Airbus A320 crash (1993)

- Cause
  - Software defect: Flight management computer delayed braking system deployment because of airplane bank angle and hydroplaning
- Effect
  - 9-second delayed deployment of spoilers and thrust reversers
  - 13-second delayed deployment of wheel brakes
  - Plane overshot runway at 72 knots (133 kph)
  - 2 deaths
- Cost
  - unknown (aircraft was destroyed)

Other expensive consequences of software failures

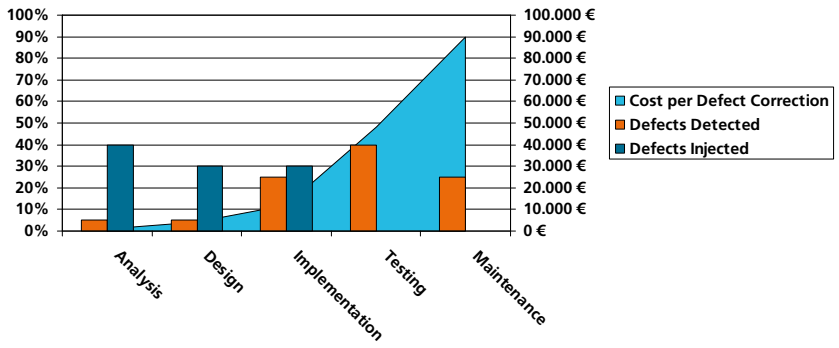
- Ariane 5 flight 501: loss of spacecraft, $370 million cost (1996)
- Hewlett-Packard: ERP system failure caused $160 million damage (2004)
- UK inland revenue: tax-credit overpayment caused $3.45 billion damage (2004/05)
- Tokyo Stock Exchange: trading suspended for about 4.5 hours because of software update (2005)
- Patriot missile: rounding error caused 28 deaths (1990)
- Mizuho: $225 million typing error (2005)
- BMW: recall of 23,000 cars with software defect (2002)
- Siemens: cellular phone Siemens S65 removed from shops (software defect caused hearing damage)
- …
- 50% (!) of all car breakdowns are caused by electronic failure (2005)

What do these disasters have in common?

- Caused by faulty software
  - Software contained defects that were not detected during development and quality assurance activities
  - Software defects led to failures of the surrounding system
- Very costly
  - Between hundreds of millions and billions of US$
  - Sometimes cost cannot even be determined
  - Loss of human lives cannot be measured in US$
  - Public humiliation

Any preventive action would have been orders of magnitude cheaper!

Costs of defect removal increase dramatically over time



Cost ratios for defect removal between development and field usage:

- IBM - 1:117
- Toshiba - 1:137

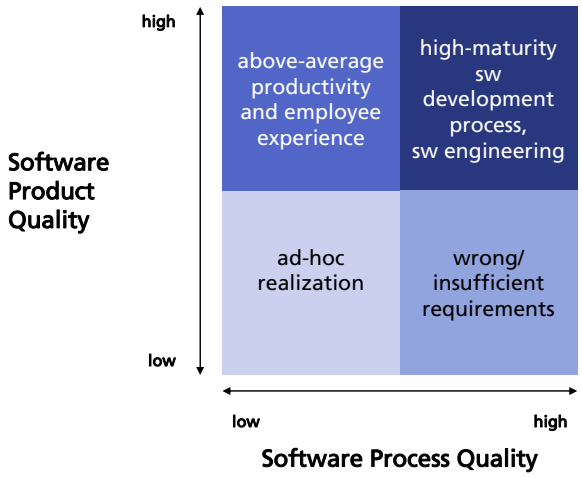
(Source: SQS AG, empirical data from 5.000 projects)

Software product failures indicate faulty processes

- Failures of software products are typically caused by defects in the software
- Defective software products indicate problems with the processes that are used for their creation
  - The development processes introduce (too many) defects into the software at the time of creation
  - The quality assurance processes are incapable of removing the necessary number of defects before delivery of the product

→ Products and processes are closely related!

Process quality vs. product quality



How can we use knowledge about defects?

- For the systematic and goal-oriented optimization of development processes and quality assurance activities
  - Prevent development activities from introducing software defects
  - Make quality assurance activities more effective (find more defects earlier) and more efficient (require fewer resources)
- To focus improvement activities according to defects in order to improve software quality most efficiently
  - Which development activity introduces most defects?
  - Which quality assurance activity finds the least defects?

→ How can the required knowledge be acquired?

Defect-flow models deliver the required knowledge

- We must know which process introduced how many defects of which severity
- We must know when these defects could be found
- We must know when these defects are actually found

- Defect-flow models track defects from their injection into the product to their removal
  - Time of injection provides information on the development process that produced the defect
  - Type of defect provides information on when it could have been found in the best case
  - Time of detection provides information on when it was actually found

Benefits of defect-flow models

- Identification of improvement potentials in applied QA approach
  - Where are which defects injected into the product?
  - Where are the fewest defect found?
  - When are certain types of defects handled (not at all | too early | too late)?
- Evaluation of process changes
  - Changes in QA activities have direct impact on defect flow
  - Deviations from baseline allow for evaluating the effects of changes
- Quality control and reduction of project costs
  - Data from already completed projects are used for estimating defect counts and defect flows expected in new projects
  - Ability to estimate defects makes quality controllable
  - Deviations from estimates can be easily identified and eliminated!
  - This contributes to a significant reduction in costs, e.g., rework costs reduced by 50-90%!