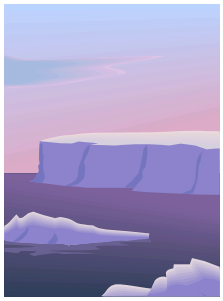
DEFECT-FLOW MODELS

PART 3: The Defect-Flow Model Approach



Michael Kläs
michael.klaes@iese.fraunhofer.de

Martin Kowalczyk
martin.kowalczyk@*iese.fraunhofer.de*

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

Ove Armbrust
ove.armbrust@iese.fraunhofer.de

Fraunhofer IESE

Fraunhofer IESE

Part 3: The Defect-Flow Model Approach

- **DFM Principles**
  - Motivation
  - Basic Principles
- **DFM Creation and Introduction Process**
  - Define Basic Model
  - DFM Motivation
  - Define Extended Model
  - Implementation
- **DFM Application**
  - Possible Applications
  - Possible Measurement Goals and Question
  - Possible Interpretation Models
- **Maintaining Defect Classification Schemes**
  - Improvement goals and cycle
  - Evaluation goals

Motivation

Prominent classification schemes

- Hewlett-Packard scheme
- Orthogonal Defect Classification (ODC)

Usage of predefined schemes

- Successful in specific contexts
- Difficult to apply in other companies' context
- Especially if documents and defects differ from traditional ones

Customized defect classification

- Fitting to specific environment
- and its measurement goals

Customized defect classification schemes

Defect-flow model approach

- Addresses properties of good classification scheme
- Process for defining, introducing, and validating customized defect classification scheme
- Interview-based approach
  - Combining expertise from measurement experts and domain knowledge from developers
- Provide measurement concept from which useful defect analyses can be selected

Defect-flow model

- Characterize development lifecycle
  - Number of injected and detected defects
  - Type of defects detected or not detected

Properties of a good defect classification scheme (1/2)

{ major,
minor,
negligible,
documentation,
unknown }

{ major,
minor,
negligible,
unknown }

{ major,
minor,
…
… } **≤ 7±2 values**

Orthogonal attributes and attribute values

- Different aspects of a defect are captured in different attributes
- At most one attribute value can be selected for each attribute

Complete attribute values

- At least one attribute value can be selected for each attribute
- If in doubt, use attribute value "other"or "unknown"

Small number of attribute values

- Numbers that are too large make selection hard and thus data unreliable
- 7 ±2 is the number of items the human brain can keep in short-term memory

Properties of a good defect classification scheme (2/2)

Clear meaning and definition of attributes / attribute values

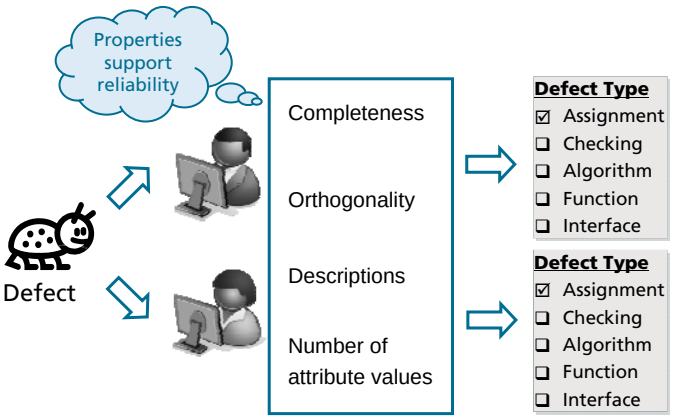
- Should be *defined in a textual way* and augmented with *examples*
- Otherwise, values might be confused, resulting in inconsistent and unreliable data

Attribute: Defect Type
Represents the actual correction that was made

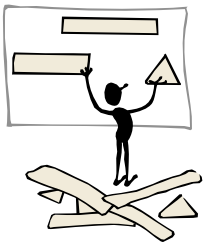
Attribute Value: Checking
Errors caused by missing or incorrect validation of parameters or data in conditional statements. It might be expected that a consequence of checking for a value would require additional code such as a do-while loop or branch. If the missing or incorrect check is the critical error, checking would still be the type chosen.
Examples:
1) Value greater than 100 is not valid, but the check to make sure that the value was less than 100 is missing.
2) The conditional loop should have stopped on the ninth iteration. But it kept looping while the counter was <= 10.

Properties and reliability

Reliability = the same defect is classified in the same way



Change Management: the introduction strategy



Management point of view

- Management commitment & data quality
- Motivate and visualize benefits of defect measurement

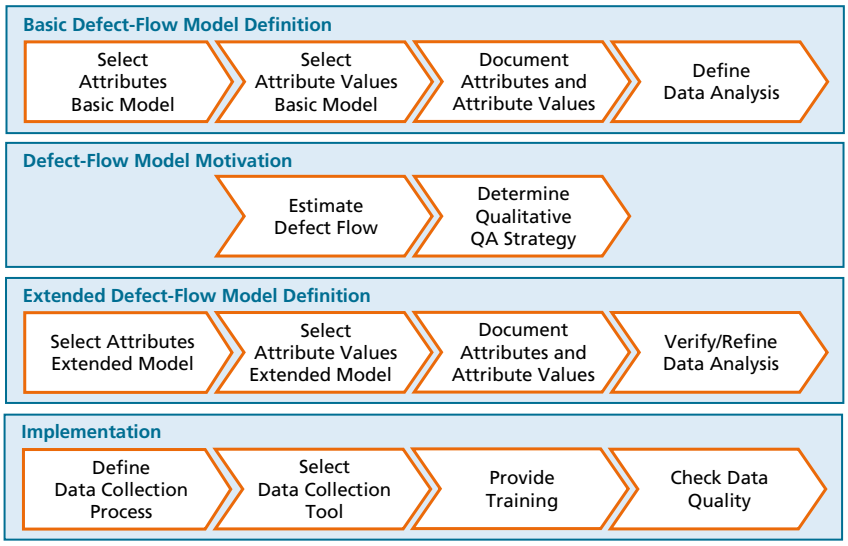
Speed of implementation

- Stepwise introduction increases acceptance
  - Collectors are not overwhelmed by the amount of new data to collected
  - Positive results in one implementation step can motivate further extension/enhancement
- Opportunities for stepwise introduction
  - Begin with basic defect-flow model
  - Refine for selected document types
  - Start with small number of projects

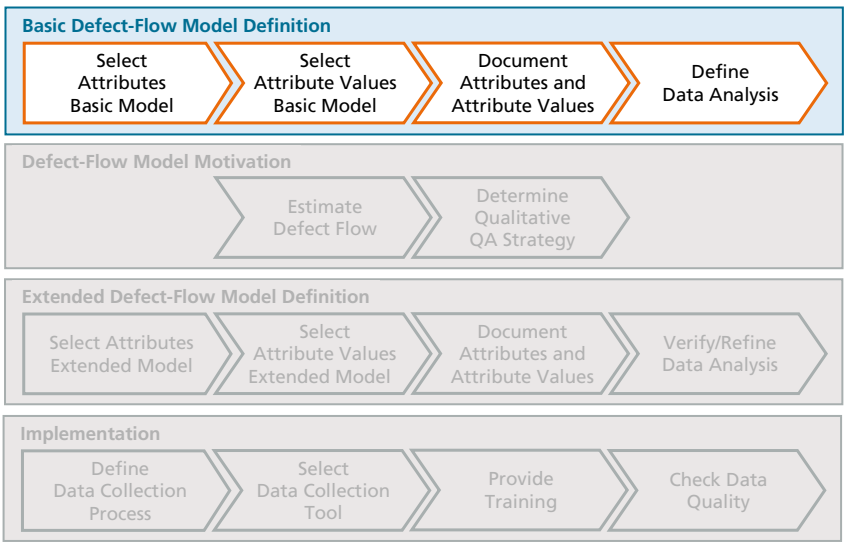
Part 3: The Defect-Flow Model Approach

- **DFM Principles**
  - Motivation
  - Basic Principles
- **DFM Creation and Introduction Process**
  - Define Basic Model
  - DFM Motivation
  - Define Extended Model
  - Implementation
- **DFM Application**
  - Possible Applications
  - Possible Measurement Goals and Question
  - Possible Interpretation Models
- **Maintaining Defect Classification Schemes**
  - Improvement goals and cycle
  - Evaluation goals

Overview: Definition and Implementation Process



Overview: Definition and Implementation Process



Phase 1 – Basic defect-flow model definition (1/5)

Select Attributes Basic Model

Select Attribute Values Basic Model

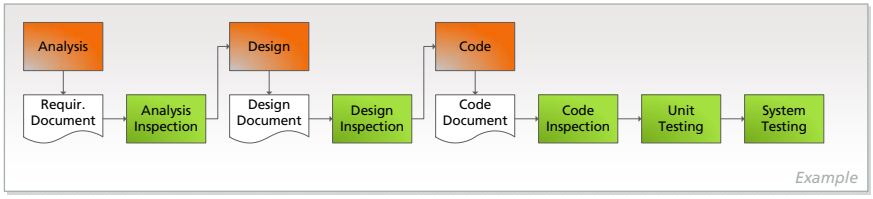
Basic Model Attributes

- Detection: activity in which the defect was detected
- Injection: activity in which the defect was injected

Attribute Values

- Derived from the process model
- Injection-detection diagram:
  - shows constructive activities, document flow,
  - and corresponding QA activities

Perform interviews: to check applicability



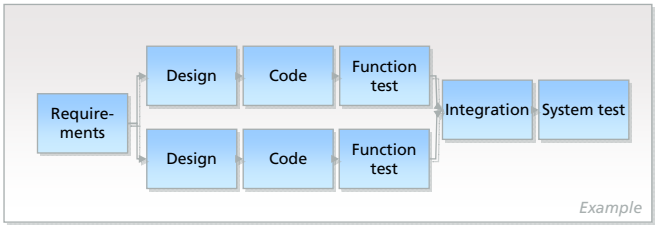
Phase 1 – Basic defect-flow model definition (2/5)

Select Attributes Basic Model

Select Attribute Values Basic Model

Optional: Consider the process topology

- Allows to investigate different branches of the product lifecycle
- Requires definition of additional attributes
  - Source of the injection / detection (e.g., department or branch of the process)



Phase 1 – Basic defect-flow model definition (3/5)

Document Attributes and Attribute Values

Attribute	*Definition*
Detection	The QA activity in which the defect was detected
*Value**	*Definition and Example*
Requirements Review	
Design Review	…

Use organization-specific terminology

Attribute	*Definition*
Injection	Phase in which the defect was injected
*Value**	*Definition and Example*
Requirements	…
Design	…
Implementation	…

** **Siemens AG*** (William A. Florac, Robert E. Park, Anita D. Carleton, Practical Software Measurement: Measuring for Process Management and Improvement., Tech. Rep. CMU/SEI-HB-003, Software Engineering Institute, Apr. 1997) *Example*

Phase 1 – Basic defect-flow model definition (4/5)

Basic Defect-Flow Model

- A defect classification scheme for characterizing the defect flow in the software lifecycle (i.e., the injection and detection of defects)

Injection: In which activity was the defect injected originally?

Requirements | Design | Code

Detection: In which activity was the defect detected?

Requirements Review | Design Review | Code Review

Function Test | System Test | Field

Example

Phase 1 – Basic defect-flow model definition (5/5)

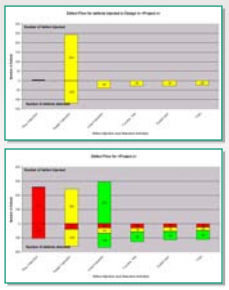
Define Data Analysis

Define concepts for data analysis …

- Define measurement goal
- Refine goal by questions
- Choose appropriate visualizations

Use, e.g., GQM

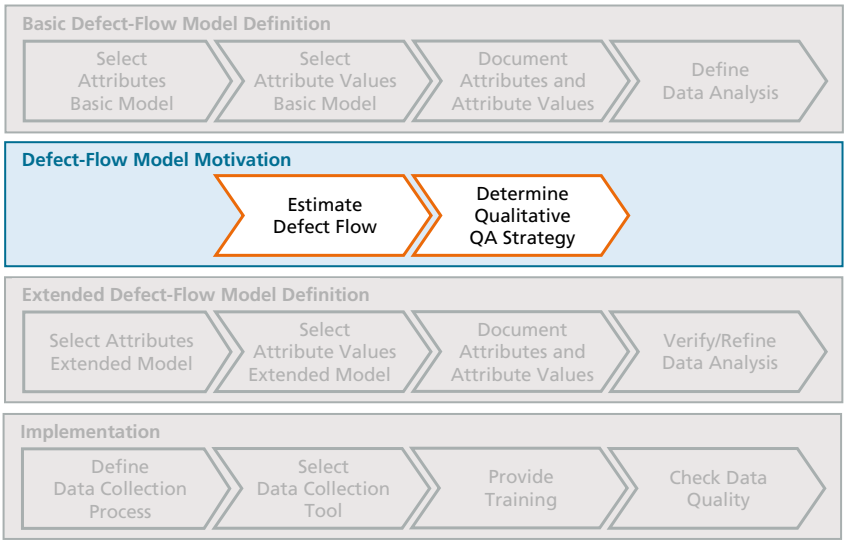
or select from provided measurement concepts



1) Characterize the effectiveness of quality gates
   - 1.1: What percentage of defects is detected in *<quality gate x>*?
   - 1.2: …
2) Characterize the defect flow throughout the development process
   - 4.1: How many defects are injected and detected throughout the development process?
   - 4.2: …

Example

Overview: Definition and Implementation Process



Phase 2 – Defect-flow model motivation

Estimate Defect Flow

Purpose: give management an initial version of how the defect flow will look like

Approach: combine project data and expert opinion to estimate the defect flow (*)

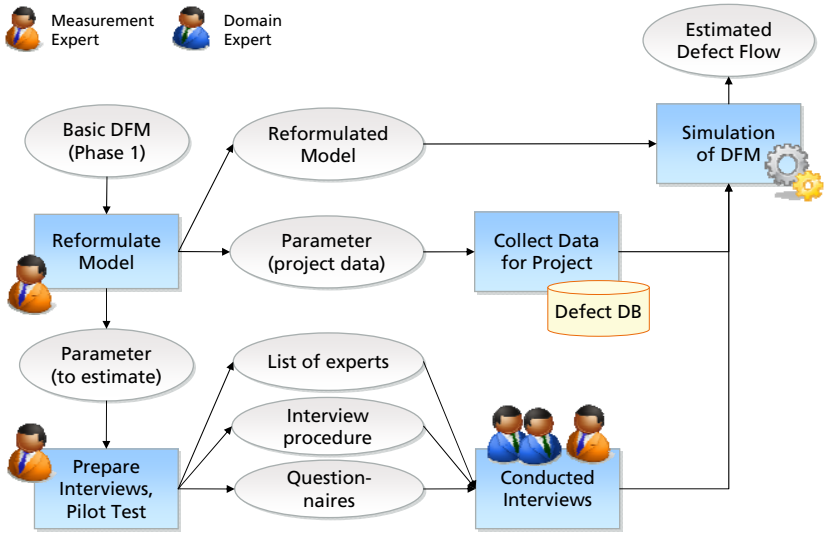
Determine Qualitative QA Strategy

Purpose: motivate implementation of more detailed defect classification for an extended defect-flow model

Approach: elicit the current qualitative quality assurance strategy

* L. C. Briand, B. Freimut, and F. Vollei, Assessing the Cost-Effectiveness of Inspections by Combining Project Data and Expert Opinion, Proc. of the 11th Int. Symp. on Software Reliability Engineering, pp. 124-135, 2000 *Example*

Phase 2 – Process for estimating defect flow



Phase 2 – Estimate defect flow

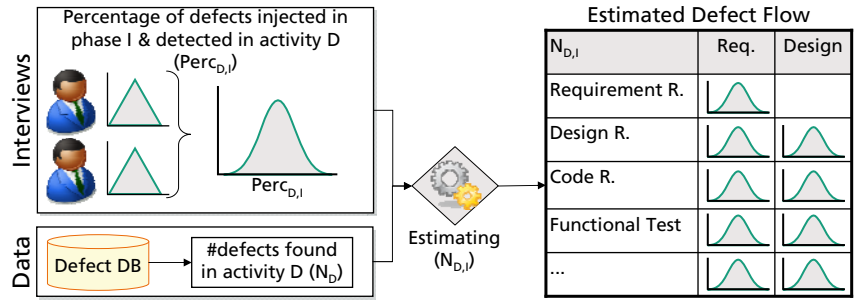
Desired information: the number of defects ($N_{D,I}$)

- detected in QA activity D and
- injected in development phase I,
- for all QA activities and development phases

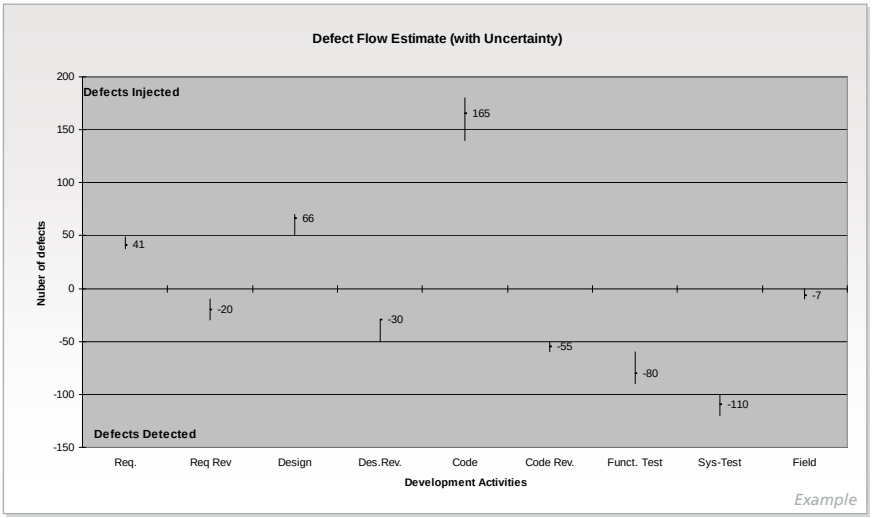
Determine which data are already available and which not

- Typically, N_D is available without information on phase of injection
- $Perc_{D,I}$ is estimated by domain experts, where $Perc_{D,I} = N_{D,I} / N_D$

Expert estimates and project data are combined using MC simulation



Phase 2 – Example of estimated defect flow



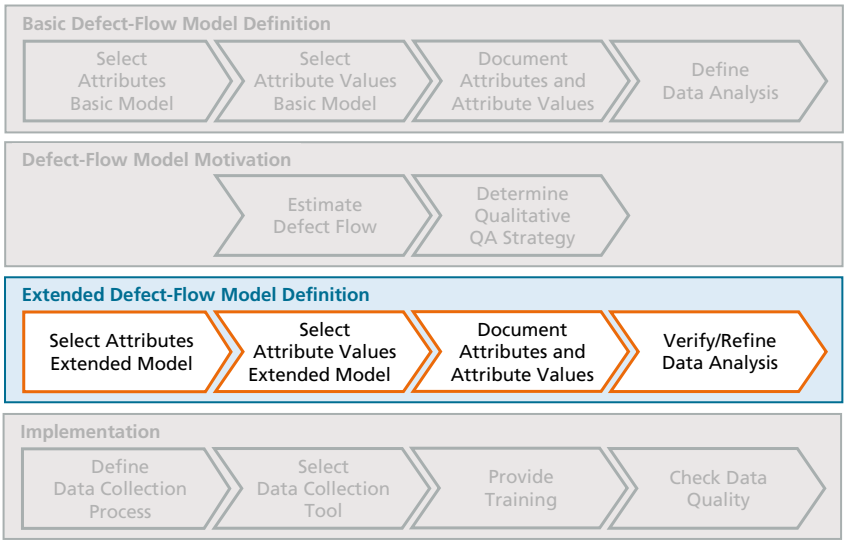
Phase 2 – Determine qualitative QA strategy

What kind of defects are hard/easy to detect in a QA activity?

Found easily	Type 1			
			Type 2	
Detection	Code Inspection	Design Review	Function Test	System Test
Found with difficulty				
	Type 2			
	Type 3			

- Completed interviews with developers
  - First visualization of the empiric QA strategy
  - May identify improvement potential
- No predefined definition of "defect type"
  - Developers should use their own, domain-specific view
  - Measurement experts learn the language of the developers

Overview: Definition and Implementation Process



Phase 3 – Extended defect-flow model definition

Select Attributes Extended Model

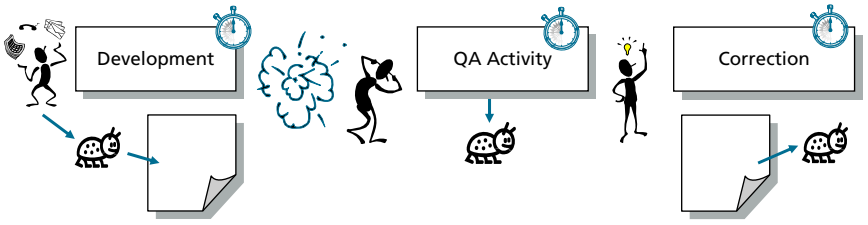
Identify relevant attributes in a goal-oriented way

- Prioritize relevant attributes
- *Correction type* is a good starting point
  - "What is fixed in order to correct the defect?"
  - Practitioners think and talk about technical defects of certain documents
  - Relating *injection* and *correction type*
- Additional attributes
  - useful for analyzing sub-parts of the data, e.g.,
    - software parts written for reuse or written for individual projects
    - code written manually or generated automatically
  - should be independent (orthogonal)

Phase 3 – Typical dimensions of defect attributes

Mechanism
How was the defect created, detected, and corrected

Timing
When was the defect created, detected, and corrected



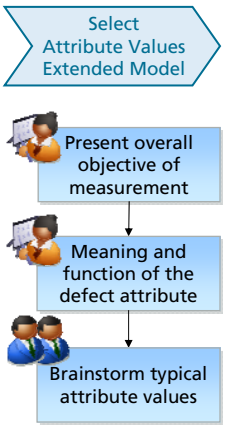
Cause
What was the error leading to a fault

Symptom
What was observed when defect surfaced or the activity revealed the defect

End Result
What is the failure caused by the fault

Location
Where in the system was the defect detected

Phase 3 – Select attribute values for extended model



Correction type values for code documents

- Start with ODC defect type (quasi-standard)

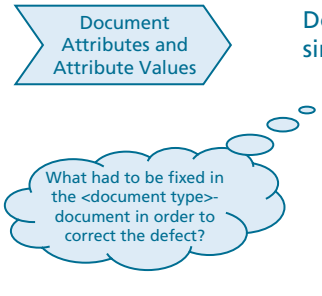
Correction type values for other documents

- Ensure that the values match the context
- Identify semantic structures a document is composed of, not only syntactic structures
- Use domain knowledge of developer

Structured interviews with small groups

- Participants: measurement expert and developer dealing with the document type

Phase 3 – Extended defect-flow model definition

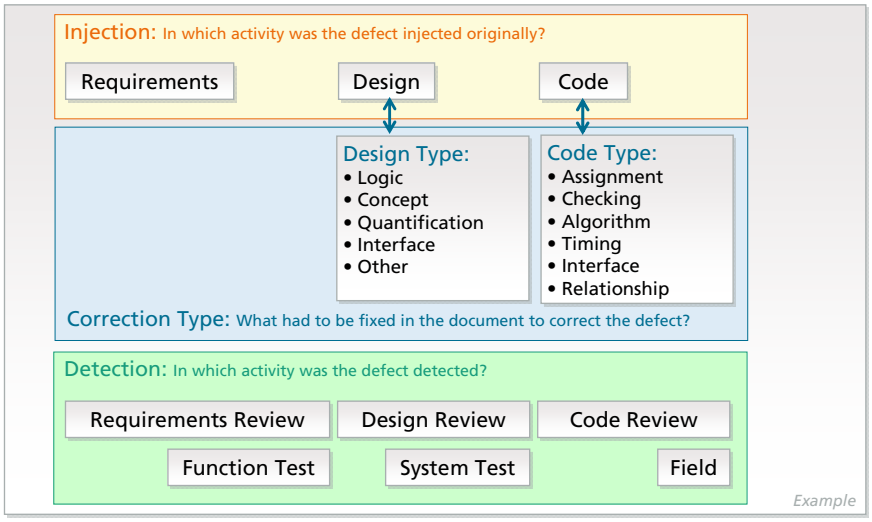


Documentation is usually performed simultaneously with attribute/value definition

- Attributes can be phrased as a question
- Attribute values can then be seen as answers to these questions
  - *name* for the attribute value
  - *definition* of when to assign a defect
  - *example* of actual defects matching definition

Verify that attribute and attribute value definition support planned data analysis

Phase 3 – Example of extended defect-flow model

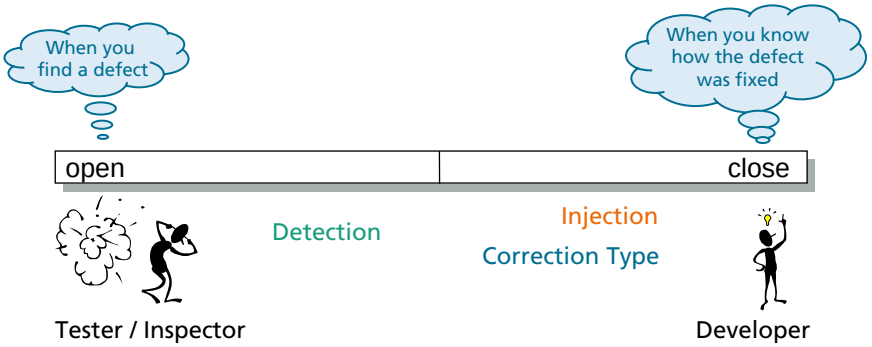


Phase 4 – Implementation of defect-flow model

Define Data Collection Process

For each attribute, determine

- who is going to collect it
- and when

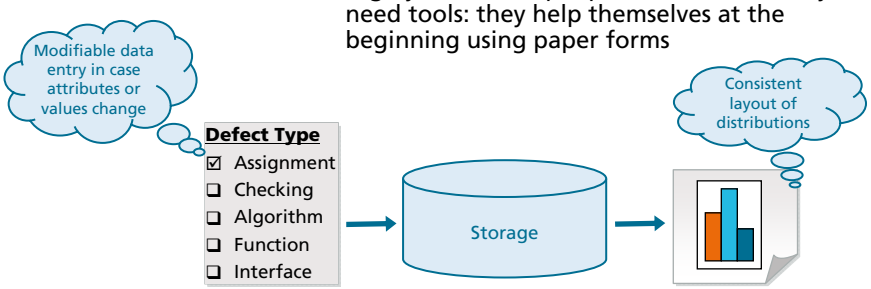


Phase 4 – Implementation of defect-flow model

Select Data Collection Tool

Determine how the data can be collected by means of an (existing) defect tracking system

- Classification itself necessary but tedious
  - provide easy-to-use, quick, simple tools
- Tool components: data entry, data storage, (support for statistical methods), graphics
- Highly motivated people do not necessarily need tools: they help themselves at the beginning using paper forms



Phase 4 – Implementation of defect-flow model

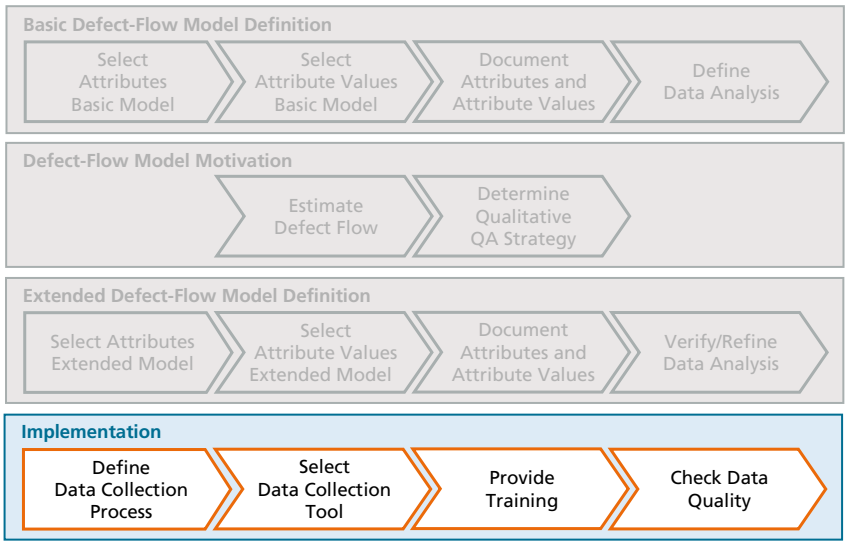
Provide Training

Provide training on the classification scheme to users and management

- Present purpose of the measurement program
  - Underline the importance for the organization
- Introduce to attributes and attribute values
  - Present developed definitions and examples

It is not enough to simply provide access to the definitions and examples!

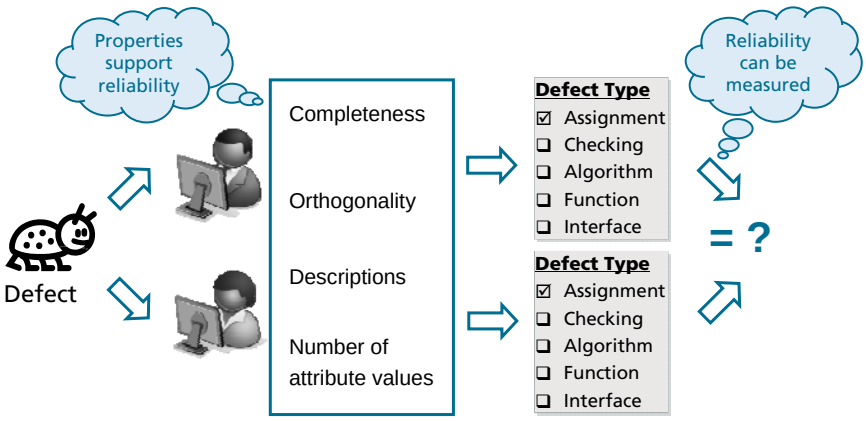
Overview: Definition and Implementation Process



Phase 4 – Check data quality (1/2)

Validate properties of scheme

- Are descriptions of attributes and their values understandable?
- Is the scheme reliable enough?



Phase 4 – Check data quality (2/2)

Validate properties of scheme

- Are descriptions of attributes and their values understandable?
- Is the scheme reliable enough?

Agreement $\kappa = \frac{P_0 - P_e}{1 - P_e}$

More in Model Maintenance Part

$P_e = \sum_{j=1}^{n} p_{j+} \times p_{+j} \quad P_0 = \sum_{j=1}^{n} p_{jj}$

Data Collector 1

Data Collector 2	Value 1	Value 2	…	Value n	
Value 1	p_{11}	p_{12}		p_{1n}	p_{1+}
Value 2	p_{21}	p_{22}		p_{2n}	p_{2+}
…					
Value n	p_{n1}	p_{n2}		p_{nn}	p_{n+}
	p_{+1}	p_{+2}		p_{+n}	

κ	Interpretation
< .4	Inadequate agreement
> .6	Good agreement
> .75	Excellent agreement

Summary – Creation and Introduction of DFM

Defect-flow models = defect classification schemes for characterizing where in the process defects are injected and removed

Main components of Basic Defect-Flow Model

- Injection and Detection

Main components of Extended Defect-Flow Model

- More detailed classification of a defect: e.g., Correction Type added
  - For code documents: e.g., customized ODC Defect Type
  - For other documents: define new attribute values

Approach to developing defect-flow models

- Distilling principles of successful classification schemes
- Interviews combining domain knowledge of developers and measurement expertise of measurement experts

Part 3: The Defect-Flow Model Approach

- **DFM Principles**
  - Motivation
  - Basic Principles
- **DFM Creation and Introduction Process**
  - Define Basic Model
  - DFM Motivation
  - Define Extended Model
  - Implementation
- **DFM Application**
  - Possible Applications
  - Possible Measurement Goals and Question
  - Possible Interpretation Models
- **Maintaining Defect Classification Schemes**
  - Improvement goals and cycle
  - Evaluation goals

Application of defect-flow model

Identification of process improvement

- Where are many defects injected?
- Where are few defects detected?
- Where are defect types not detected?

Baseline for evaluating process changes

- Process changes (esp. on the quality assurance) have a direct impact on the defect flow

Control the quality budget of development projects

- Measure the number of detected defects in project x
- For a similar project x+1: scale for project size and predict expected number of defects
- Interpret deviations between expected and actual number of defects

Measurement concept – example DFM questions (1/2)

G1: Characterize the effectiveness of the quality gates in the development process.

- **Q1.1**: What percentage of defects is detected in <quality gate x>?
- **Q1.2**: What kind of defects are detected in <quality gate x>?
- **Q1.3**: What kind of defects slip though <quality gate x>?
- **Q1.4**: In which quality gates do we detect <defect type y>?

G2: Derive process improvement actions in order to find defects earlier or more efficiently (i.e., improve defect detection).

- **Q2.1**: What percentage of defects is detected in <*quality gate x*>?
- **Q2.2**: What kind of defects slip though <*quality gate x*>?

Measurement concept – example DFM questions (2/2)

G3: Derive process improvement actions in order to prevent the occurrence of systematic defects (i.e., defect prevention).

- **Q3.1**: What activity introduces many defects?
- **Q3.2**. What kind of defects is introduced most often?

G4: Characterize the defect flow throughout the development process.

- **Q4.1**: How many defects are injected and detected throughout the development process?
- **Q4.2**: In which quality gates do we detect <*defect type y*>?

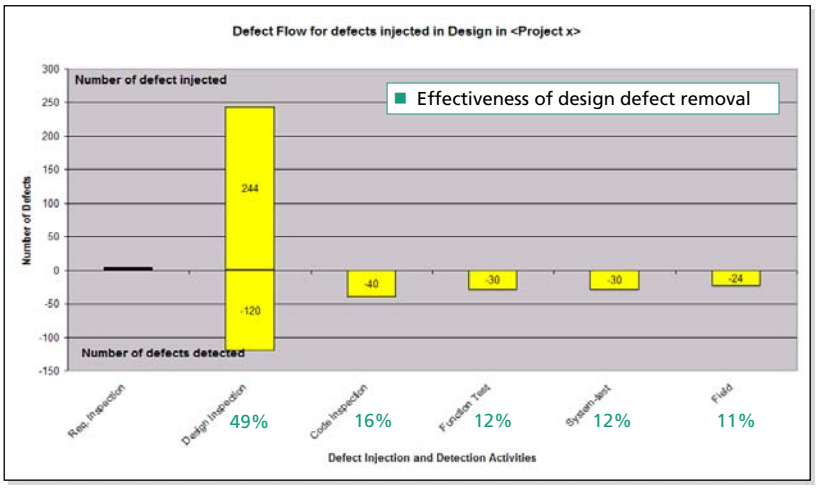
G5: Predict the number of defects occurring in a development project.

- **Q5.1**: How many defects will be detected in the development process?

Next → Define measures and interpretation models

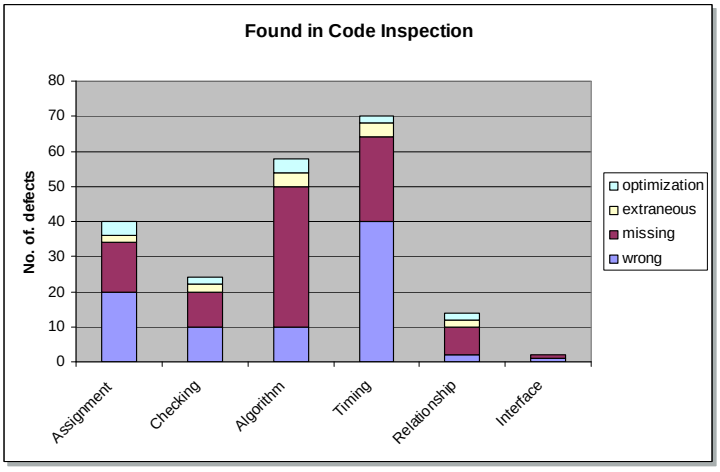
Measurement concept – DFM interpretation models (Q1.1)

- **Q1.1/2.1**: What percentage of defects is detected in <gate x>?



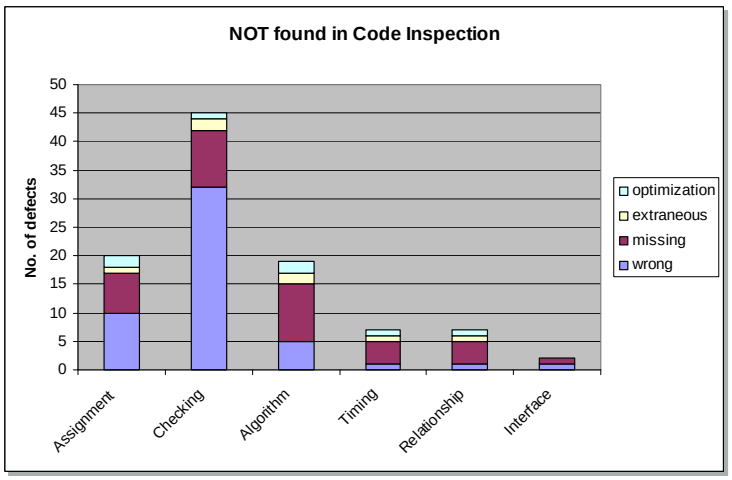
Measurement concept – DFM interpretation models (Q1.2)

- **Q1.2**: What kind of defects are detected in <quality gate x>?



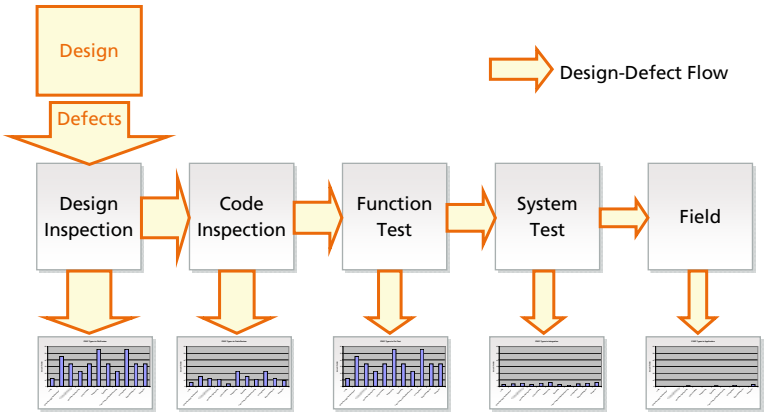
Measurement concept – DFM interpretation models (Q1.3)

- **Q1.3/2.2**: What kind of defects slip through <quality gate x>?



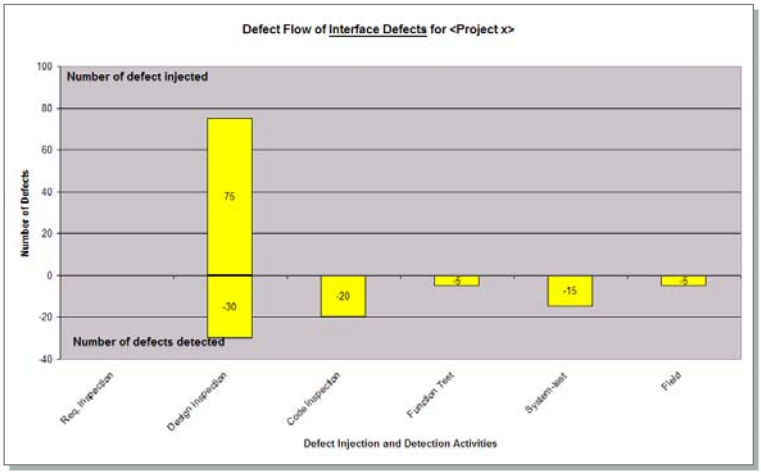
Measurement concept – DFM interpretation models (Q1.4) (1/2)

- **Q1.4/4.2**: In which quality gates do we detect <defect type y>?



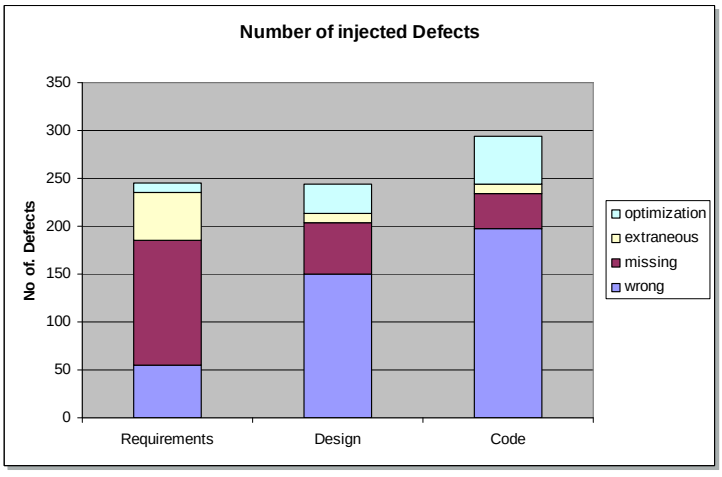
Measurement concept – DFM interpretation models (Q1.4) (2/2)

- **Q1.4/4.2**: In which quality gates do we detect <defect type y>?



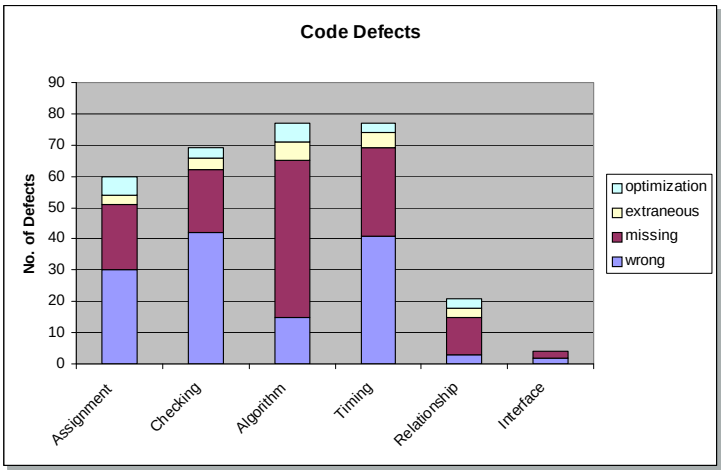
Measurement concept – DFM interpretation models (Q3.1)

- **Q3.1**: What activity introduces many defects?



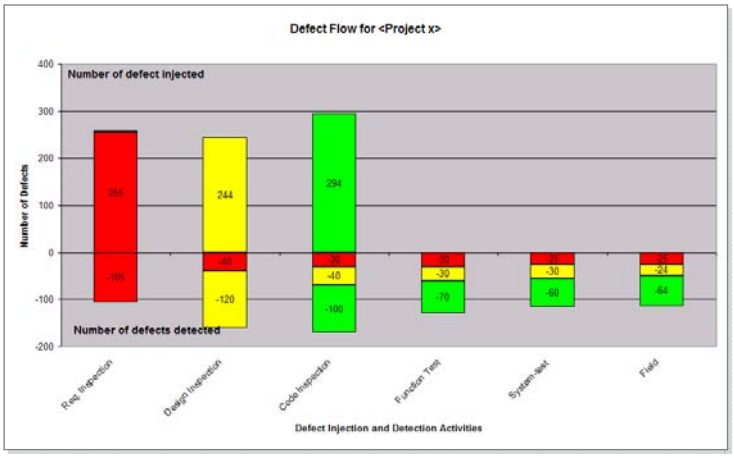
Measurement concept – DFM interpretation models (Q3.2)

- **Q3.2**: What kind of defects is introduced most often?

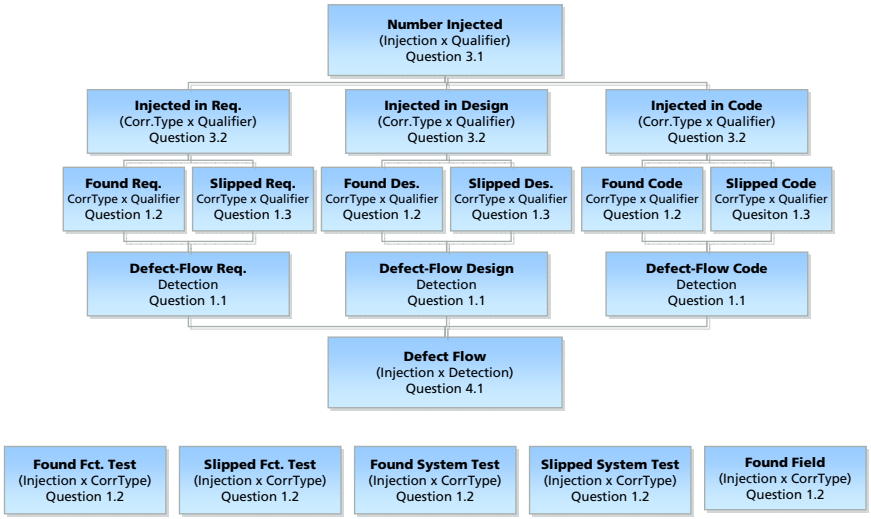


Measurement concept – DFM interpretation models (Q4.1)

- **Q4.1**: How many defects are injected and detected throughout the development process?



Measurement Concept – Overview of Analyses



Part 3: The Defect-Flow Model Approach

- **DFM Principles**
  - Motivation
  - Basic Principles
- **DFM Creation and Introduction Process**
  - Define Basic Model
  - DFM Motivation
  - Define Extended Model
  - Implementation
- **DFM Application**
  - Possible Applications
  - Possible Measurement Goals and Question
  - Possible Interpretation Models
- **Maintaining Defect Classification Schemes**
  - Improvement goals and cycle
  - Evaluation goals

Improving the performance of defect-flow models

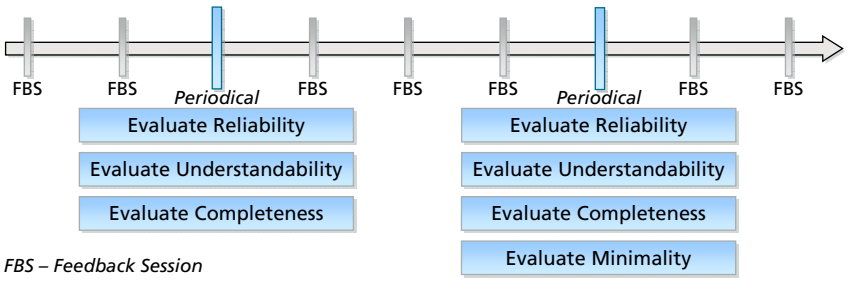
General Goal: Improve performance of the defect-flow models

Specific Goals

- Collected measurement data are complete and valid (correct)
- Defect-flow models are up to date
- Defects are classified reliably
  - Reliability of defect classification scheme
  - Understandability of defect class definitions
  - Completeness of defect classification schema
  - Minimality of defect classification schema

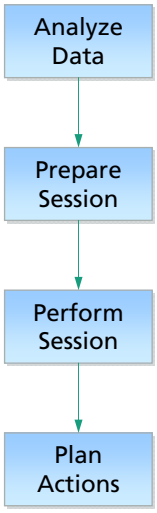
Example cycle of DFM maintenance

- Continuous improvement (on a daily basis): approx. effort = 2-3 days per year
  - Validation of the collected measurement data
  - Trainings and feedback sessions for developers and managers
- Periodical improvements (every 6 or 12 months): approx. effort = 5-7 days per year
  - Different maintenance activities
  - Maintenance cycles depends on the maturity of an organization (e.g., stability of processes)



FBS – Feedback Session

Example means for maintaining DFM – Feedback Sessions



- Perform standard data analysis
- Identify critical values
- Analyze defect data
- Prepare open questions
- Perform session
- Collect additional comments, opinions
- Plan concrete improvement actions
- Consolidate with QA

Method: Combine data analysis and expert judgment

Goal: Identify improvement potentials in the DFM

Procedure

- Involve measurement and development/domain experts
- Present goals and critical questions
- Present results of data validation and analysis
- Collect feedback
  - Subjective evaluation of the DFM process
  - Discuss possible sources of deficiencies identified in data validation and analysis
  - Discuss potential solutions to identified problems (and their possible causes)

Part 3: The Defect-Flow Model Approach

- **DFM Principles**
  - Motivation
  - Basic Principles
- **DFM Creation and Introduction Process**
  - Define Basic Model
  - DFM Motivation
  - Define Extended Model
  - Implementation
- **DFM Application**
  - Possible Applications
  - Possible Measurement Goals and Question
  - Possible Interpretation Models
- **Maintaining Defect Classification Schemes**
  - Improvement goals and cycle
  - Evaluation goals

Evaluating reliability

- Goal: Evaluate goodness and reliability of defect classification scheme
- Time frame: every 6 months
- Method: determine Cohen's Kappa coefficient
  - Evaluate consistency of multiple defect classifications
  - Compare classifications of the same set of defects provided by two independent experts
  - The value of the Kappa coefficient quantifies the level of agreement between the two classifications

Evaluating reliability – procedure (2/3)

- Select sample set of defects from the historical data base (>= 50 defects)
  - Possibility A: Statistical sampling (random sample)
  - Possibility B: Convenience sampling (i.e., the *N* most recently detected defects)
- Collect the defect sample in the form of a table:

Defect ID	Defect Description	Defect Classification by Expert
Defect 1	…	…
Defect 2		
…		
Defect N		

Evaluating reliability – procedure (2/3)

- Select value for the attribute "defect class" (e.g., type of correction)
- Compare classifications originating from different sources (e.g., provided by different raters)
  - Transfer the classification data to the agreement/disagreement evaluation table

		Developer 1										
		Logic	Quantis	Fkt-Erkl	Mod.Richt	Doku	Init	Applik	Task	Konzept	Schnittst	Resource
Developer 2	Logic	5			1	1						
	Quantis		3				1					
	Fkt-Erkl			5								
	Mod.Richt		1		5							
	Doku		1		3	6						
	Init						3					
	Applik					1		2				
	Task											
	Konzept				1					2		
	Schnittst	1			1						1	
	Resource				1							1
		6	5	5	12	8	4	2	0	2	1	1
iet, in	Misclass.	3	3	0	8	6	1	1	0	1	2	1

- Note: An analysis is also possible with more than 2 experts (Fleiss' Kappa).

Evaluating reliability – procedure (3/3)

- Compute Cohen's Kappa coefficient (e.g., using an Excel sheet)

$$\text{Agreement } \kappa = \frac{P_0 - P_e}{1 - P_e} \qquad P_0 = \sum_{j=1}^{n} p_{jj} \quad P_e = \sum_{j=1}^{n} p_{j+} \times p_{+j}$$

- Evaluate the value of the Kappa coefficient

Kappa κ	Interpretation (Landis and Koch 1977)
< 0	Poor agreement
0.0 – 0.20	Slight agreement
0.21 – 0.40	Fair agreement
0.41 – 0.60	Moderate agreement
0.61 – 0.80	Substantial agreement
0.81 – 1.00	Almost perfect agreement

Source: J.R. Landis, G.G. Koch, "The measurement of observer agreement for categorical data" Biometrics, vol. 33, 1977. pp. 159-174.

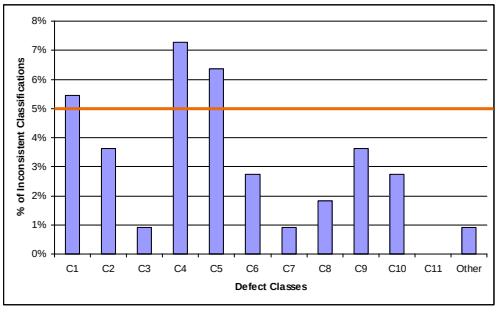
- Based on the result, plan appropriate actions
  - Review and modify classification schema w.r.t. classification consistency

Evaluating understandability

- Goal: Evaluate understandability of defect class definitions
- Time frame: every 6 months
- Method
  - Based upon the reliability evaluation according to Cohen's Kappa coefficient
  - Exact analysis of the definitions of defect classes for which inconsistent classifications were provided
    - i.e., for which experts disagreed in their classifications when classifying a sample set of historical defects

Evaluating understandability – procedure (1/2)

- Analyze the results of the evaluation with respect to "disagreements" between individual defect classifications (Cohen's Kappa coefficient) provided by independent experts
  - Focus on the defect with the highest percentage of inconsistent classifications



- Analyze definitions of defect classes whenever more that 5% of the defects have been classified inconsistently

Evaluating understandability – procedure (2/2)

- Analyze the definitions of affected defect classes
- Perform interviews with those developers who classified defects differently than specified in a reference classification
- Based on the interview results, derive appropriate actions for improving consistent understanding of the defect classification scheme
  - Provide additional training with respect to unclear defect classes (i.e., classes that were the most frequent sources of misclassifications)
  - Update defect classification by modifying definition and/or attributes and/or attribute value for selected defect classes
  - Define additional defect classes (if necessary)
  - Provide training for new and updated defect classes

→When updating defect classification, remember to

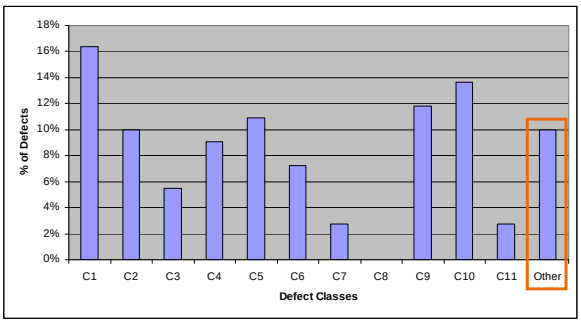
  - Consider and maintain defect classification quality attributes (e.g., reliability, minimality, etc.)
  - Involve members of quality assurance team
  - Map the existing classification onto the new one

Evaluating completeness

- Goal: Assure completeness of the defect classification scheme
- Time frame: every 6 months
- Method
  - Evaluate defects that were classified as "other"
  - Check if any new defect class can be derived for classifying the "other" type of defects
  - Check if any existing defect class can be refined (e.g., definition and examples) to cover the "other" type of defect

Evaluating completeness – procedure (1/3)

- Analyze the distribution of defect classes collected during the last 6 months



- If ratio of defect classified as "other"
  - < 3%, then there is no need to react
  - >= 3%, then current defect classification scheme should be analyzed for possible extension in order to cover the "other" type of defects

Evaluating completeness – procedure (2/3)

- Analyze the free-text descriptions attached to the defects that were classified as "other"
  - Goal: Gain an understanding of these defects and look for their potential commonalities
  - If the number of defects is too high, consider a limited sample, e.g., 20-30 of the most recently detected defects
- Consider the results of the consistency evaluation (values of Cohen's Kappa coefficient)
- Perform interviews with developers who classified defects as "other"

Evaluating completeness – procedure (3/3)

- Cluster defects according to their common characteristics
  - Based on the results of interviews and free-text descriptions
- Assess if clusters indicate any new defect classes
  - Yes → discuss with QA staff the possibility of introducing new defect classes (updating defect classification scheme)
  - No → defects must remain in the "other" class
- If necessary: Update defect classification and plan appropriate trainings

When updating defect classification, remember to

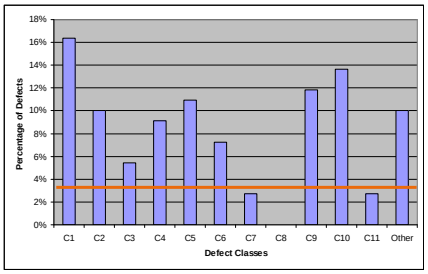
- Consider and maintain quality of defect classification
  - e.g., reliability, minimality, etc.
- Involve members of quality assurance team

Evaluating minimality

- Goal: Identify potentially redundant defect classes
- Time frame: every 12 months
- Method
  - Evaluate defect classification charts from last 12 months
  - Identify and discuss with developers and QA staff those defect classes
    - that have hardly been employed or
    - that have not been employed at all

Evaluating minimality – procedure (1/2)

- Analyze distribution of defects across available defect classes in the considered time frame (i.e., last 12 months)



- Focus on the defect classes for which the ratio (or the absolute number) of classified defects lies below a certain threshold

Evaluating minimality – procedure (2/2)

- Perform interviews with developers
  - Goal: Find out potential rationale behind observed distribution of defects across defect classes
- Perform interviews with QA staff
  - Goal: Find out if defect classes that are rarely used (or not used at all) are really/actually relevant
- Based upon the outputs of the interviews, appropriate actions should be planned with respect to the rarely used defect classes
  - If defect classes are relevant or if there are other reasonable rationales behind them remaining in the defect classification scheme
    - → No need for actions
  - If defect classes are not relevant and there are no additional rationales for their existence
    - → Remove defect class from classification scheme
    - → In the future, classify defects of this type as "other"
    - → Update defect classification scheme and plan appropriate trainings