Controlling Project Performance by Using the Project Defect Model.
Ben Linders, Ericsson, The Netherlands

1 Introduction

Wouldn't it be nice if you could have more insight into the quality of a product, while it is developed, and not afterwards? Would you like to be able to estimate how many defects are inserted in the product in a certain phase, and how effective a (test) phase is in capturing these defects? The simple but very effective model described in this paper makes it possible! The model is used at Ericsson to develop software for telecommunications product. It supports controlling projects, by putting quality next to planning and budget, evaluating risks, and taking decisions regarding release and maintenance.

This paper will first highlight why there was a need for such a model, and why existing measurements didn't fulfill this need. Then the model itself, and the deployment in the projects are described. Conclusions that were drawn from the model, using feedback sessions, are described, explaining how the projects have benefited from the model. At the end we look shortly into the future, regarding both the model and the needs of the organization regarding measurements on product quality.

2 Why a Model?

Within Ericsson there has always been focus on the quality of developed products, next to planning and budget. Initially measurements like fault density were used. But fault density has major drawbacks; one being that you can only measure it after a phase is completed, and another is that it does not give any insight on the causes if a product has a quality risk. For instance, high fault density can either mean that there is a quality risk, that the product was more thoroughly tested than other products, or both. The same applies for a low fault density, the reason could be that insufficient testing was done and that defects remain undetected in the product (a product quality risk), or that the product has a better quality and thus less defects were found, or both. Studies outside of Ericsson have also revealed the limited value of fault density; see for instance [1], other studies showed defect measurements that successful organizations used [2]. So there was a need for new measurements that would give more insight. The GQM metric approach was used to define the measurements [3].

Goals:

1. Control verification activities (optimize defect detection).
2. Control development activities (minimize defect injection).
3. Predict release quality of the product.
4. Improve the quality of the development and test processes.

There is need for measurements usable to steer quality: measurements to plan quality at the start of the project, and track it during project phases. Enabling corrective and preventive actions and reducing quality risks in a project. An additional projects need is to estimate the number of latent defects in a product at release. The purpose is twofold. In the first place, it is usable to decide if the product can be delivered to customers, or released, knowing the quality. Secondly, it helps to plan the support and maintenance capacity needed to resolve the defects that are anticipated to be reported by customers. Finally it should be possible to have quality data that is analyzed together with the applied processes, and the way a project is organized. This analysis provides insight into process and organizational bottlenecks, and therefore enables cost efficient improvements.

Questions:

1. What will be the quality of the released product?
   1.1. Per requirement?
   1.2. As perceived by customers?

2. How good are inspections?
   2.1. How effective is the preparation?
   2.2. How effective is this review meeting?
3. How good are the test phases?
   3.1. How many test cases are needed?
   3.2. How effective is a test phase?
4. What is the quality of the requirement definition?
5. What is the quality of the high level and detailed design?
6. What is the initial quality of the code (before inspections/test)?
7. Which phase/activity has the biggest influence on quality?

This list of questions is not exhaustive, but they are the first ones that come to mind when you want to measure and control quality. Certain questions can trigger additional questions, for instance when it appears that a certain test phase is ineffective in finding defects, additional questions are needed to investigate the activities and its effectiveness.

Metrics:

1. Number of undetected defects in the released product.
2. Number of defects found per requirement/feature.
3. Number of latent defects in a product before an inspection or a test phase (available).
4. Number of defects expected to find in an inspection.
5. Actual number of defects found in an inspection (detected).
6. Number of defects expected to be found in a test phase.
7. Actual number of defects found in a test phase (detected).
8. Size of the document/code.
9. Detection rate: percentage of defects detected (detected/available).

The metrics listed above can be collected in most projects, since the data is usually available in inspection records and defect tracking tools. But to analyze the metrics, a measurement model is needed. This since the metrics are related, only when looking at a combination of several metrics conclusions can be drawn that help answering questions and reaching the goals of the measurements.

3 How does the Model Look?

To get more insight into the quality of the product during development, the software development processes must be measured with two views: Introduction and detection of defects. To develop the model, descriptions from Watts Humphrey [4] and Stephen H. Kan [5] have been used.

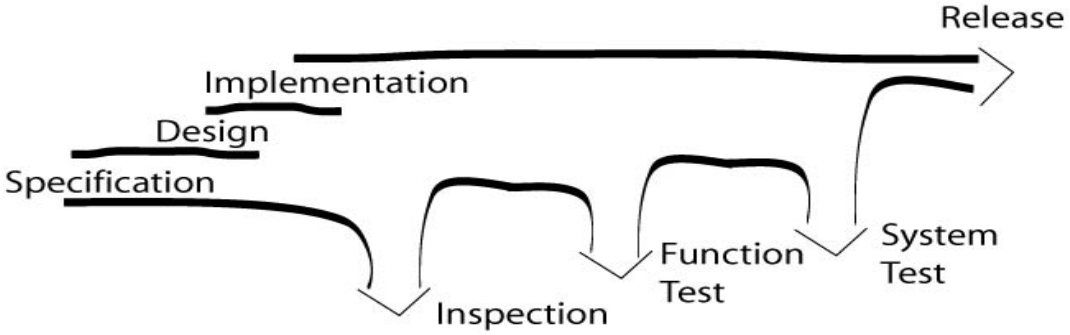


Figure 1: Defect Flow

Introduction is done during the specification, design and coding phases; defects are either introduced into documents or into the actual product. Measuring introduction gives an indication of development

phase quality. Detection of defects is done via inspections and test during all the phases of the project. Measuring detection gives insight into the effectiveness of verification phases. By using these two measurements, a project can determine if there is a quality risk, and what the origin is: Too many defects in the product and/or insufficient testing to capture the defects.

3.1 Development Phase Quality

The quality of a product depends on the number of defects that are inserted during the development phases. Mistakes are made in every phase, from specification to implementation. Defects that are detected and removed increase the likely quality of the end product. However, those defects reflect the inefficiency of the development process. Defects which are not detected in the phase in which they are inserted lead to more and expensive rework and can decrease product quality if they remain in the product after release and surface when customers use the product. The aim is to remove defects as early as possible and to have as few defects as possible in the product when released, thereby delivering quality products.

At the start of a phase the number of inserted defects is estimated. During execution of the phase this estimate is adjusted based on the number of defects actually detected. Since it is sometimes difficult to estimate the number of defects, an alternative method is to estimate the size of the produced documents or code, and use size multiplied by the defect density to estimate the number of defects. In all cases, it is better to do a rough estimate, and adjust it during a project, than to do no estimate. Historical data of earlier projects is very useful when estimating defect introduction. Also industry data is used when no historical data is available.

3.2 Defect Detection Effectiveness

The aim of verification is to detect the inserted defects, preferably in the earliest phase that they can economically be detected. The effectiveness is expressed with a detection rate, that is:

Detection Rate = Number of defects detected / Number of defects present in product

An organization has a detection rate for a certain phase, which is estimated within certain statistical limits. Initially when no historical data of an organization is available, industrial figures can be used. An alternative for the detection rate is to estimate the absolute number of defects that are likely to be found in the current phase. Based on that number and the number of defects present, the detection rate is derived.

During the execution of a phase, the detection rate is adjusted based on the actual number of defects detected. If, for instance, a detection rate of 50% is expected, and 46% of the expected defects are detected halfway through the phase, then either the number of defects that was inserted will be higher than initially expected, or the actual detection rate is higher – fewer defects were inserted than were predicted. If the first is true, then there is a quality risk in the product, which needs to be investigated. Also it gives a signal usable to improve the process phase where the defects were introduced. In a next increment of the project, defect introduction can thus be reduced. If fewer defects were inserted and thus the resulting detection rate is higher, then further investigation is warranted to understand how this was accomplished. That would make it possible to learn and improve verification in other projects, based on the positive experiences from this one.

The combination of measurements on defect insertion and defect detection gives a more detailed view of the quality of the product, and effectiveness of the development processes. This provides a project with better means to track and control quality.

4 Deployment of the Model

4.1 Pilot Project

The defect introduction and detection model as described in the earlier paragraphs is implemented in a pilot project for a network management product. Since the project has two distinct requirements, the project is divided in two increments with separate teams, which are overlapping in time. The model copes with these two increments separately, since different processes are used. As the first part of the project was combined for the two increments, and also final testing was combined, the basic introduction/detection model becomes:

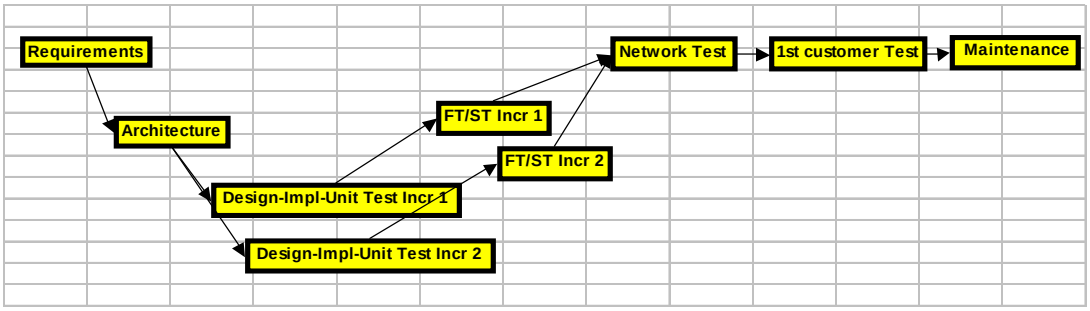


Figure 2: Project Phases

A tool for the model is developed using a spreadsheet: The Project Defect Model. The purpose of the Project Defect Model is to estimate defects inserted and detected by phases, and to track defects from inspections and tests against the estimates. The model supports analysis of the data with both calculated values and graphs comparing actuals to estimates in terms of current status and trends.

In the pilot, 420 defects are collected, which are analyzed and classified on introduction phase, requirement, and phase where they could have been detected. The result data gives an estimate of 21 latent defects in the released product, expected to be found in the first six months of operation. This estimate was used as one of the criteria in the release decision; it was decided that this would be an acceptable quality level provided that sufficient maintenance support would be available to solve the 21 defects when detected by customers. The 6 months operation period ended in June 2003, and 20 defects were actually found, a difference of 1 defect with the estimate at release.

Activity	Latent defects	Estimated defects #	%	
Inspection	420	197	47%	Plan
Test in project	223	194	87%	Test
MDA/FOA	24	9	38%	Only %
Project totals	420	400	95%	91%
Maintenance	20	20	100%	
Average/Total		420		

Figure 3: Defect Figures from Pilot Project

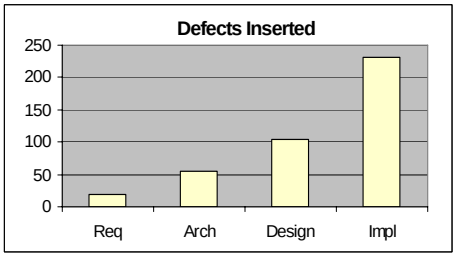


Figure 4: Defect Inserted per Phase

Based on the estimated number of latent defects, the project has a defect detection rate of 95%, i.e. 400 of the 420 defects made in the project have been detected before the product is released. If we exclude the phases before test (that used inspections for verification) from the measurement, the detection rate is lower: only 91% of the defects left after inspections were detected in the test phases. This shows that inspection has contributed towards the quality of the released product. However, the average detection rate from inspections is 47%. According to industry data, inspections can detect between 60%-80% of the available defects, so there is room for improvement.

Even more important than the data are the benefits the project received by using the model. During the project, data feedback and analysis sessions are done where corrective actions based on the data are implemented Major conclusions/actions included:

- A slip through of requirement defects is detected early in the architecture phase. Investigation showed however that good high-level design, combined with effective architecture inspections, revealed many requirement clarifications. Action defined is to monitor requirement defect detection in the design phase for quality risks; it turned out that both the number and impact of the detected defects are limited. No more requirement defects are detected in later phases, final conclusion is that the requirements after initial clarification reached a high quality.

- Data from defects inserted/detected, test requirement coverage, and Orthogonal Defect Classification, shows that inspection effectiveness depends on several issues: Good and focused inspectors, qualified moderators, sufficient preparation, and thorough inspection planning. The detailed conclusions on inspections are used to further improve reviews and inspections in future projects. Though it was known that inspections are an effective way of detecting defects (as was to be expected from many earlier studies), our data confirmed this and has lead to more focus from management and buy in for further improving inspections.

- Data shows that test phases discover defects that could have been found earlier. Function Test finds many inspection defects, where System Test discovers a lot of Function Test defects. Based on Trigger analysis with Orthogonal Defect Classification, we determine test progress. Together with a requirement based test matrix, the project predicts where requirements are sufficiently verified, and where there are risks of latent defects. Test focus and scope is changed during the project, based on data; the remaining quality risks are on requirements that are seldom used.

The Project Defect Model is beneficial to the project. It helps estimating, planning, and tracking quality during the project. This quality data is used in the project together with time and cost data, to take better decisions. Also the model identifies quality risks at an early stage, helping the project to take corrective actions and decisions on product release and maintenance capacity planning. The teams using the model gain significant quantitative insight into their design and test processes, that they will use in future projects. Feedback sessions of defect data analyzed by the team themselves prove to be very powerful.

More detailed information about the model and the results from the pilot can be found in [6].

4.2 Data from finished and ongoing projects

Based on the results in the pilot project, the management team has decided that all future projects would use the Project Defect Model to estimate and track their quality. Until now (march 2005) there are 11 projects from R&D in Rijen, that have used or are using the model. Also the model is used to do retrospective analysis on some older projects, to get data to derive planning constants for future projects. Below data collected from the projects:

Project detection rates Q1 2005 (PSQT Conference)												
	Proj A	**Proj B**	**Proj C**	**Proj D**	**Proj E**	**Proj F***	**Proj G**	**Proj H***	**Proj J**	**Proj K**	**Proj L**	**Average**
Rate	95%	95%	90%	59%	97%	86%	93%	88%	91%	94%	93%	91%
Size	1	4	1	1	5	3	1	4	1	2	3	

The detection rate calculates which percentages of the defects are found within the project, before delivery to the customer. Size is a relative indication on how big the project (man-hours and lead-time) is. This table shows that on average, 91% of the defects are detected in the project, while the customers detect 9% of the defects. Project D was a project which integrated earlier developed and tested components, and only did function and system test, therefore the lower detection rate.

Our projects range from 86% to 97%. Industry figures for "best in class" vary between 90% and 95% (see [7]). Based on that and current performance, the target for our organization is set on 90%-95%. This implies that we track and analyze projects that find too little defects, but also projects that find too many defects. The first is obvious, and we are analyzing projects F and H who are currently not meeting the target. Corrective actions have been defined, and the expectation is that will meet the target before release. Projects that are finding too many defects could be performing "too good", with the risk of testing too much. We have started a pilot with Cost of Quality to measure and optimize test costs, and support test decisions.

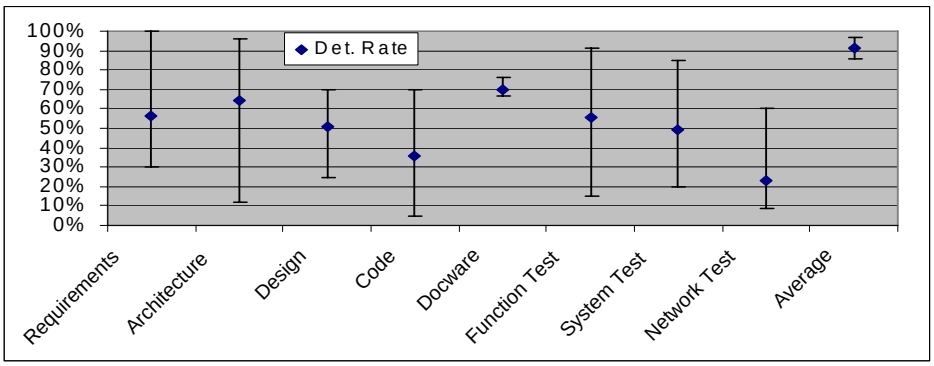
Phase injection rates, Q1 2005 (PSQT Conference)					
	Requiremen	Architectur	Design	Code	Docware
Rate	**7%**	**18%**	**12%**	**58%**	**4%**

These figures show what kinds of defects are made in projects. We see that most of the defects are coding defects, while architecture and design are the 2nd and 3rd biggest categories. Given the fact that much effort is put in exploring, defining and verifying the architecture, that includes formal inspections on architecture documents, this is an expected and wanted result.

Phase detection rates, Q1 2005 (PSQT Conference)									
	Requiremen	Architectur	Design	Code	Docware	Function Te	System Tes	Network Te	Average
Det. Rate	**56%**	**64%**	**51%**	**36%**	**70%**	**56%**	**49%**	**23%**	**51%**

The phase detection rate calculates which percentage of the available defects in a product at the start of a phase is captured in that phase. We see that the requirements, architecture and design inspections have high detection rates, while code inspections rate is lower. An improvement program is ongoing to increase the effectiveness of code inspections. Docware (customer documentation) inspections have a high detection rate due to a strong inspection team with broad product and customer knowledge.

Function test and system test both have a good performance. Network test is lower, but the kind of defects that are found would have given significant problems to customers. So finding them in test improves product quality before release to the customers. The average figure of all detection phases is 51%, given industry figures this acceptable but there is room for improvement.



The picture above shows the variance in detection rates for the projects. In general there is a significant variance, except for docware. Main reason is that projects differ in the inspection and test strategies that are used, deploying them to their specific needs. The variance in the total project detection rate is however much smaller, as we saw earlier. But still, we are investigating possibilities to decrease differences in test processes and methods (decreasing variance), and we are deploying best practices (increasing the average).

The figures above from all projects help us to define planning constants that are used for future projects, thereby improving our estimation accuracy.

4.3 Feedback sessions

Organizations are increasingly relying on measurements, but many struggle to implement them. There are the usual technical problems associated with collecting data, storing it efficiently, and creating usable reports. However, the biggest challenges are often related to using the data to actually make decisions and steer the activities of the organization. Miscommunication, incomplete analysis, and corrective actions that seem to come from nowhere create resistance to the whole idea of measurements.

Feedback is based on the assumption that you should give the raw data to the people who did the work, and that they should perform the analysis. Why? Because they know the story behind the data. For instance, defect detection rates are discussed with the test team leader, he knows how much and what kind if testing they have been doing, and they expect to find.

With the Project Defect Model we do regular feedback sessions. On average once a week we look at the data, compare it to our estimates, and check where there are differences, trends, or signals in the data that something is going wrong. This is compared with the development status from the design and test teams, based on that we draw conclusions and take the necessary actions.

We see that development teams learn a lot from the feedback they received on defects. They see which kinds of defects they discover too late, and use the data to improved the early test and inspection processes. For instance, for defects that slip through many times, checks are add to the design and inspection checklist. Using these checks the defects are found before or at the latest in inspection. One project found analyzing the data that both product knowledge and test skill are the cause that insufficient defects are found in early test. The test team now takes time to study product behavior documentation, and uses coaches to support newcomers on the team. As a result, the detection rate of the test phase increases significantly, thus reducing the number of defects available in the product when delivered to system test. So during the improvements they use the data from the model to check if they are actually making progress.

Also the teams get insight where they make the most defects, using the data they are able to determine root causes and improve quality right at the start. In a project a team found out that a specific part of the project is more complex and difficult to verify. They put in extra time in the investigation of possible solutions, to prevent many small but disturbing defects during design and coding, and reduced the risk that defect would slip through to late testing.

An effective feedback process doesn’t come easily. In the beginning it will need a lot of attention and perseverance, but once the benefits of the effort become clear, which is usually early in the process, people will start to give their support. More information about feedback in measurement systems, including the key success factors & pitfalls, can be found in [8].

5 Conclusions

The Project Defect Model is beneficial for our projects. It helps estimating, planning, and tracking quality during the projects. This quality data is used in the projects together with time and cost data, to improve decision-making. The model identifies quality risks at an early stage, helping the projects taking corrective actions and decisions on product release and maintenance capacity planning. Also the design and test teams using the model gain significant quantitative insight into their design and test processes, that is used in future projects.

Currently the model is extended with effort spend in design and test phases. This enables trade-off between appraisal cost (pre-release defect detection), rework cost (pre-release defect removal) and operational cost (post-release defect removal), evolving into a Cost of Quality model.

Note:

An earlier version of this paper has been published in the NESMA Anniversary book:
Measure! Knowledge! Action! ISBN: 90-76258-18-X www.nesma.org

About the Author

Ben Linders is Specialist Operational Development & Quality at Ericsson Telecommunicatie B.V., the Netherlands. He has a Bachelor in Computer Science, and did a Master study on Organizational Change. He works in process- and organizational improvement for more then 15 years, implementing high maturity practices, to improve organizational performance and bring business benefits.

Since 2000 he leads the Defect Prevention program. He coaches implementation of Root Cause Analysis, Reviews and Inspections, and has defined and applied a Project Defect Model, used for quantitative management of the quality of products and effectiveness of verification.

Also he introduces and supports the measurement system, manages continuous improvement, and is an expert and coach in several Organizational Performance & Quality areas.

Ben is the president of the SPIder (Dutch SPIN) foundation, and a member of several (national and international) SPI and quality related networks. He has written several papers, and regularly gives presentations. You can reach him at +31 161 24 9885, email ben.linders@ericsson.com.

References

[1] A Critique of Software Defect Prediction Models, Norman Fenton and Martin Neil. In IEEE Transaction on Software Engineering, September/October 1999.

[2] Software Measurement Programs and Industry Leadership. Capers Jones. In Crosstalk February 2001. http://www.stsc.hill.af.mil/CrossTalk/2001/feb/jones.asp

[3] The Goal/Question/Metric Method: A Practical Guide for Quality Improvement of Software development. R.v.Solingen, E.W. Berghout, McGraw-Hill.

[4] Managing the software process. Watts Humphrey. Chapter 16, managing Software Quality.

[5] Metrics and models in Software Quality Engineering. Stephen H. Kan. Chapter 6, Defect removal effectiveness.

[6] Controlling Product Quality During Development with a Defect Model. Ben Linders. In: Proceedings of the 8th European SEPG conference, London, 2003.

[7] A Business Case for Software Process Improvement Revised, DACS state of the art report. Measuring Return on Investment from Software Engineering and Management.

[8] Make what's counted count, how one company found a way to use measurements to steer the actions of their organization. Ben Linders. In: Better Software magazine, march 2004.

For more information, contact:
Ben Linders
Operational Development & Quality
Ericsson Telecommunicatie B.V.
Research and Development
Tel: +31 161 24 9885
Fax: +31 161 24 2617
ben.linders@ericsson.com
http://www.ericsson.nl