A study of inspections and testing at Ericsson, Norway

Reidar Conradi, Amarjit Singh Marjara, Øivind Hantho,
Torbjørn Frotveit, and Børge Skåtevik

Inspections and testing represent core techniques to ensure reliable software. Inspections also seem to have a beneficial effect on predictability, total costs and delivery time.

This paper presents a case study of inspections and testing, done at the Ericsson development department outside Oslo, Norway. This department developed and maintained customer-defined services around AXE phone switches, i.e. the functionality around the "star"" and "square" buttons on house telephones.

AXE development at Ericsson in Norway, used a simple, local experience database to record inspections and testing data. The historical data used in this study is from the period from 1993 to 1998. Three diploma (MSc) students from NTNU have been given access to historical data in the years 1997-99. The results from the three diploma theses constitute the basis for the paper.

The paper will study questions such as:
- *The effectiveness and cost-effectiveness of inspections,*
- *The cost-effectiveness and defect profile of inspection meetings vs. individual reading,*
- *The relation between complexity/modification-rate and number of defects in different development phases,*
- *Whether defects found in a module during Design Inspections can be used to predict defects in the same module in later phases and releases.*

The work was carried out as part of the Norwegian SPIQ project on software process improvement in 1997-99. The paper is a rewritten version of the one presented at PROFES'99 (Oulu, Finland, 22-24 June 1999). It is extended with data from a longitudinal study, i.e. tracing modules across phases and releases.

Introduction

The paper presents results from three MSc theses at NTNU in 1997-99 [Marjara97] [Skaatevik99] [Hantho99]. These have analyzed historical defect data at Ericsson in Oslo, Norway -- related to their AXE switches. Ericsson has practised Gilb inspections [Gilb93] for many years, and collects defect data from inspections and testing in a small database. The work was carried out as part of the Norwegian SPIQ project on software process improvement in 1997-99 [Dybå00].

These studies support the view that Design Inspections indeed are the most cost-effective verification technique. Design Inspections tend to catch 2/3 of the defects before testing, by spending 10% of the development time and thereby saving about 20% of the time (by earlier defect correction, a ``win-win"). However, we have no data on the types of defects found during different development phases, e.g. whether the "easy" defects are found during inspections, and the "hard" ones during tests.

Inspection meetings were also cost-effective over most testing techniques, so they should not be omitted. Inspection meetings also found the same “type” of defects (Major, Super Major) as individual inspections.

We also found that there is a significant correlation between module complexity/modification-rate, and number of defects found during Field Use, but not during Design Inspections and test. Due to missing data, we could not clarify whether the number of defects per modules repeated itself across inspection/test phases and over several releases. That is, we could not predict ``defect-prone'' modules. However, defect classification was in general unsatisfactory and prevented analysis of many interesting hypotheses.

The paper is organized as follows: Section 2 summarizes some relevant parts of the state of the art, especially of inspections. Section 3 first describes the Ericsson context, and section 4 describes questions and hypotheses for the study. Section 5 describes the organization of the study, and Section 6 presents and discusses the results. Section 7 sums up the paper and recommends some future work.

1.State of the art

Reliability is of crucial importance for most software systems. Common remedies are sound methods for system architecture and implementation, high-level languages, formal methods and analysis, and inspection and testing techniques. Especially the latter two have been extensively described in the literature, and a vast empirical material has been collected, analyzed and published. This paper only refers to general test methods, so we will not comment further on these here.

Inspections were systematized by Fagan [Fagan76] [Fagan86] and represent one of the most important quality assurance techniques. Inspections prescribe a simple and well-defined process, involving group work, and have a well-defined metric. They normally produce a high yield, i.e. by spending 10% of the development time, we diagnose 2/3 of the defects before testing, and save over 20% of the total time – thus a win-win; quality is ``free''. Inspections can be applied to most documents, even requirements [Basili96]. They also promote team learning, and provide general assessment of reviewed documents.

Current research topics are:

The role of the final inspection meeting -- emphasized by Tom Gilb [Gilb93], but see also [Votta93]?

What is the optimal yield (cost/benefit) for inspections?

When to stop testing?, cf. [Adams84].

The effect of root-cause-analysis on defects.

The role of inspection vs. testing in finding defects, e.g. their relative effectiveness and cost. [Lott96]?

The relation between general document properties (e.g. complexity) and defects.

The relation between defects in different lifecycle phases and across releases.

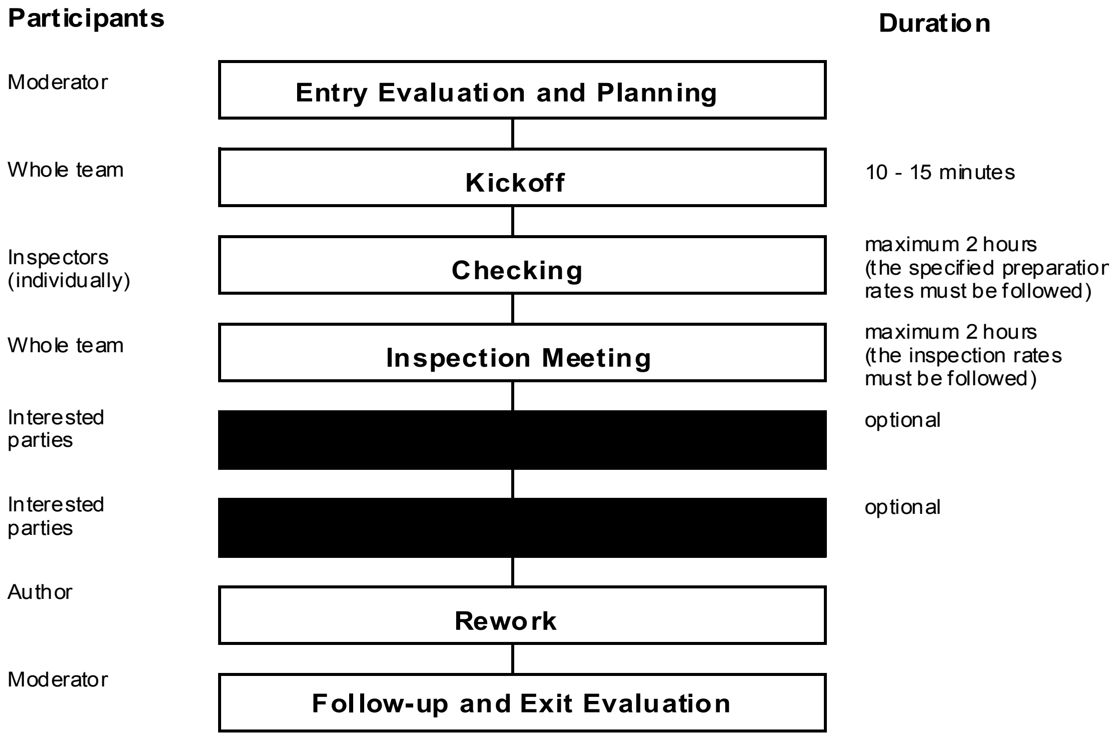
Our research questions and hypotheses will deal with the three latter.

2.The company context

Ericsson employs about 100,000 people world-wide, whereof 20,000 in development. They have company-wide and standardized processes for most kind of software development, with adaptations for the kind of work being done.

In the AXE project, Ericsson has used a classic waterfall model, with "tollgates" at critical decision points. In all this development, verification techniques like inspections and testing are crucial. Inspection is done for every life-cycle document, although we will mostly look at design and code artifacts. Testing consists of Unit Test (emulator test of SDL code), Function Test and System Test, where the two latter may be done at some integration site (e.g. Stockholm) different from the development site (e.g. Oslo).

In this paper we will only study Design Inspections (in groups), simplified Code Reviews (as part of so-called Desk Checks, by individuals), and partly testing.


Participants
Duration
Moderator
Entry Evaluation and Planning
Whole team
Kickoff
10 - 15 minutes
Inspectors (individually)
Checking
maximum 2 hours (the specified preparatior rates must be followed)
Whole team
Inspection Meeting
maximum 2 hours (the inspection rates must be followed)
Interested parties
optional
Interested parties
optional
Author
Rework
Moderator
Follow-up and Exit Evaluation

The first level inspection process

Figure 1 Basic inspection process at Ericsson for design artifacts (documents).

Figure 1 shows the inspection process at Ericsson. It is based on techniques originally developed by Michael Fagan [Fagan76] at IBM and refined by Tom Gilb [Gilb93]. It follows Fagan/Gilb wrt. overall set-up, duration etc. The number of inspectors per document is typically 3-4. Special check-lists are used for each document type. The process is tailor-made by the local development department. In addition, there is a simplified Code Review done by individual developers (data from Code Review and Unit Test are sometimes merged into a "Desk Check"). Thus full inspections are only done upon design documents in our studies. Data from inspections/reviews and testing are collected in a simple, proprietary database and used for local tuning of the process. Defects are classified as being Major, SuperMajor and Questions (the latter is omitted here) -- thus no deep categorization. We have studied software development at the Ericsson site outside Oslo. It just passed CMM level 2 assessment in Oct. 1998, and have continued to work with the CMM model. The Oslo development site has about 400 developers, mostly working on software. The actual development department, which does not exist anymore, had about 50 developers, and worked mostly on the AXE-10 digital-switch software. This switch contains many subsystems, where each subsystem contains a number of modules. The development technology was SDL as a design language and their proprietary PLEX language from the late 1970s (with own compilers and debuggers) for implementation. Special inspection groups are formed, called product committees (PC), to take care of all impacts on one subsystem. In this paper, we will only look at subsystem-internal inspections, not across subsystems.

Table 1 shows the different types of documents that are checked in the inspection process. Each document type has specific recommended inspection rates [Skåtevik99], see Table 6 later.

Document type	Application Information
ADI	Adaptation Direction
AI	Application Information
BD	Block Description
BDFC	Block Description Flow Chart
COD	Command Description
FD	Function Description
FDFC	Function Description Flow Chart
FF	Function Framework
FS	Function Specification
FTI	Function Test Instruction
FTS	Function Test Specification
IP	Implementation Proposal
OPI	Operational Instruction
POD	Printout Description
PRI	Product Revision Information
SD	Signal Description
SPL	Source Parameter List
SPI	Source Program Information

Table 1. Document types (18 different types were used).

3.Presentation of Observations, Questions and Hypotheses

3.1One Observation: O1

Observations come from simple, explorative data analysis, with no predefined questions or hypotheses. We have one resulting observation:

O1: The (cost-effectiveness) of inspections and testing.

3.2Three Questions: Q2-Q4

Questions are pre-formulated, but not so formal as hypotheses. We have three questions:

Q2: Are inspections performed at the recommended inspection rates?

Q3: How cost-efficient are the inspection meetings?

Q4: Are the same kind of defects found in individual reading as in final inspection meetings?

3.3Three Hypotheses: H5-H7

For each question we present one null hypothesis, $\mathbf{H_0}$, which is the one that will be tested and an alternative hypothesis, **HX** (X=4..7), which may be considered valid if the null hypothesis is rejected. For the statistical tests presented in this paper, a significance level (p-level) of 0.10 is used.

We will present three main hypotheses. In each case, the null hypothesis will represents the negative conclusion (no significant correlation), and the alternative hypothesis will conclude with the opposite. The three hypotheses are:

H5: There is a significant correlation between number of defects found during Field Use and module complexity/modification-rate.

H6: There is a significant correlation between number of defects during Unit Test and module complexity.

H7: For individual documents/modules, there is a significant correlation between number of defects across phases and between defect densities across releases. -- i.e. can we track and partly predict "defect-prone" modules? Six sub-hypotheses **H7.1-H7.6** are formulated in section 6.7.

4.Organization of the study

We have performed three studies where we have collected and analyzed historical data from software department at Ericsson in Oslo. Torbjørn Frotveit, our middleman at Ericsson, has all the time furnished the team with the requested data.

This paper presents results from these two studies of inspection and testing:

- **Study 1:** This is the work done in a diploma thesis from the autumn 1997 [Marjara97]. Marjara investigated inspection and test data from Project A of 20,000 man-hours (14 man-years). Defect data in this work included Design Inspections, Desk Check, Function Test, System Test, and partly Field Use. *Note:* Marjara had worked at Ericsson for five years between a regional college education and his NTNU studies. He therefore knew the company, the people and the process. In other words, he was a crucial person in initiating all these three studies.

- **Study 2:** This is the follow-up work done in a diploma thesis from the autumn 1998 [Skåtevik99]. This thesis has data from 6 different projects (Project A-F), including the project Marjara used in Study 1. It represents over 100.000 man-hours (70 man-years). The test data in this work include only data from Design Inspections and Desk Check, since later testing was done by other Ericsson divisions. However, it was possible to split Desk Check in Code Review and Unit Test, and data from these to activities are presented. Data from Field Use are not included, due to same reasons as for Function- and System Test.

- **Study 3:** This is a follow-up work from Study 2 in a diploma thesis from 1999 [Hantho99]. This thesis uses the same data set as described in Study 2.

Threats to internal validity:

We have used standard indicators on most properties (number of defects, inspection rates, effort consumption etc.), so all in all we are on agreed ground. However, wrt. module complexity we are unsure, and further studies are needed. Whether the recorded defect data in the Ericsson database are trustworthy is hard to say. We certainly have discovered some inconsistencies and lots of missing data, but our confidence is still pretty high.

Threats to external validity:

Since Ericsson has fairly standardized working processes world-wide, we can assume at least company-wide relevance, although the processes are now undergoing change. However, many of the findings are in line with previous empirical studies, so we feel confident on a general level.

5.The results and their evaluation

This chapter presents the results from the three studies described previously in section 5, and concludes tentatively on the questions and hypotheses stated in section 4.

Three definitions will be used throughout this section:

Effectiveness: the degree to which a certain technique manages to find defects, measured by number of diagnosed defects in a program (unit), regardless of cost.

Defect density: effectiveness **(**number of defects) divided by some volume measure, usually number of document pages.

Cost-effectiveness or **efficiency:** effort to find one defect (i.e. effort/effectiveness), often in person-hours per defect.

5.1O1: The (cost-)effectiveness of inspections and testing

This section describes and compares the effectiveness and cost-effectiveness of inspections and testing at Ericsson in Oslo. The effort spent *before* individual reading is proportionally distributed over inspection reading and inspection meetings. The effort of the inspection phase being spent *after* inspection meetings, is similarly merged into "defect fixing" (see Figure 1). Table 2 is taken from Study 1 and shows the effectiveness of inspections and testing, All efforts are in person-hours, sometimes just called hours.

Activity	Defects [#]	[%]
Inspection preparation, design	928	61.8
Inspection meeting, design	29	1.9

Desk Check (Unit Test + Code Review)	404	26.9
Function Test	89	5.9
System Test	17	1.1
Field Use	35	2.3
Total	1502	100.0

Table 2. Effectiveness: Total defects found, Study 1.

Table 2 shows that inspections are the most effective verification activity, finding almost 64% of total defects found in the project. Second best is the Desk Check that finds almost 27%. We can also see that only about 3% of the defects found by inspections are found in the inspection meetings. Function Test and System Test found only 7% of the total defects, reflecting that most defects were found by earlier defect detection activities.

To analyze which of the verification activities that are most effective, the time spent on each activity was collected. Table 3 shows the time spent on the six verification activities.

Activity	Defects [#]	Total effort on defect detection [h]	Cost-effectiveness [h:m per defect]	Total effort on defect fixing [h]	Estimated saved effort by early defect removal ("magic formulae") [h]
Inspection reading, design	928	786.8	00:51	311.2	8200
Inspection meeting, design	29	375.7	12:57		
Unit Test and Code Review	404	1257.0	03:07	-	-
Function Test	89	7000.0	78:39	-	-
Total so far	1450	9419.5	6:19	-	-
System Test	17	-	-	-	-
Field Use	35	-	-	-	-

Table 3. Effectiveness, effort, and cost-effectiveness of inspection and testing, Study 1.

When combining effort and number of defects, Design Inspections proved to be the most cost-effective. Not surprisingly, Function Test is the most expensive activity (note: we have no effort data on System Test). It should be noted that only human labor is included for Desk Check (see below) and Function Test. The costs of computer hours or special test tools are not included. Neither is the human effort spent in designing the test cases.

In Study 2 it was not possible to get defect data from Function Test, System Test and Field Use (representing 9.3% of the defects in Study 1). Instead, the data allowed splitting of Desk Check into Unit Test (emulator test of design) and Code Review. Table 4 shows the results.

Activity	Defects [#]	[%]
Inspection reading, design	4478	71.1
Inspection meeting, design	392	6.2
Unit Test, design	598	9.5
Code Review	832	13.2
Total	6300	100.0

Table 4. Effectiveness: total defects found before Function Test, Study 2.

Again, the data show that inspections are highly effective, contributing to 77% of all the defects found before Function Test. Code Review is second best, finding almost 13% of such defects. Compared to Study 1, there is also an "efficiency improvement" in the inspection meeting, whose effectiveness has increased from 3% to 8% for defects found during inspections alone.

Table 5 shows the effort (person-hours) of the activities from Study 2. In this study, no data from Function Test or later tests were available.

Activity	Defects [#]	Total effort on defect detection [h]	Cost-effectiveness [h:m per defect]	Total effort on defect fixing [h]	Estimated saved effort by early defect removal ("magic formulae") [h]
Inspection reading, design	4478	5563	01:15	11737	41500
Inspection meeting, design	392	3215	08:12		
Unit Test, design	598	4388	07:20	-	-
Code Review	832	2440	02:56	-	-
Total	6300	15606	02:29	-	-

Table 5. Effectiveness, effort, and cost-effectiveness on inspection and testing, Study 2.

The inspection meeting itself is more cost-effective in Study 2 (8h:12min per defect) than in Study 1 (12h:57min per defect).

In Study 2 covering 100,000 person-hours, a total of 20,515 person-hours were spent on inspections (including 11,737 person-hours on defect fixing). It has been calculated ("magic formulae") that inspections did save 41,000 person-hours, which would have been necessary to locate and correct defects otherwise found by later testing. That is, a net saving of 21% of the total project effort (however, excluding effort for System test).

Study 1 covered 20,000 person-hours where 1474 person-hours were spent on inspections (including 311.2 person-hours on defect fixing). In this study it was calculated that Ericsson saved 8200 person-hours, or a net saving of 34%! However, Study 2 did not include cost data from Function Test, as did Study 1, so the actual saving must be lower than 34%.

5.2Q2: Are inspections performed at the recommended inspection rates?

We also wanted to check if the recommended inspection rates (person-hours [h] per document) were applied in the inspections. Regrettably, the available material contains only total inspection rates, covering the entire inspection process from planning to defect fixing. The results are presented in Table 6. Note also, that one document type (SPI) from Table 1 is omitted, and we do not have relevant data for all documents. Thus, the below sums of 15536 hours and 3904 defects are lower than those in Table 5.

According to the recommended rates in Table 6, the total inspection process is performed too fast, as 15536 hours are spent on inspections, whereas 19769 hours are recommended expenditure. The defect average per page is 0.43.

Study 1 also concluded with even more deviating results, as only 54% (1474 actual person-hours out of 2723 recommended person-hours) are totally used during inspections including defect fixing. The results of Study 1 are based on the same document types as for Study 2, but for fewer projects.

Document information				Actual	Recommended		Defects	
Document type	Number of documents	Total number of pages	Average length of document [page]	Actual time [h]	Planning constant [h/doc]	Recom-mended time [h]	Total number of defects	Defect density (#defects per page)
ADI	1	7	7.00	36	20	20	12	1.71
AI	29	241	8.31	1019	72	2088	197	0.82
BD	41	1038	25.32	1438	40	1640	468	0.45
BDFC	54	3376	62.52	3531	104	5616	802	0.24
COD	4	31	7.75	105	14	56	38	1.23
FD	33	1149	34.82	2432	38	1254	784	0.68
FDFC	19	897	47.21	1230	26	494	338	0.38
FF	14	366	26.14	868	20	280	363	0.99
FS	14	244	17.43	950	24	336	205	0.84
FTI	2	605	302.50	216	14	28	22	0.04
FTS	2	154	77.00	840	14	28	44	0.29
IP	3	65	21.67	257	15	45	73	1.12
OPI	5	61	12.20	130	20	100	14	0.23
POD	4	23	5.75	116	20	80	29	1.26
PRI	57	582	10.21	1651	96	5472	399	0.69
SD	4	59	14.75	300	18	72	47	0.8
SPL	27	141	5.22	417	80	2160	69	0.49
Total	**313**	**9039**		**15536**		**19769**	**3904**	**0.43**

Table 6. Planned versus actual inspection-time consumption, Study 2.

As reported in other literature, plots on inspection rates and defect detection rates (see Figure 2) show that the number of defects found per page decreases as the number of pages (document length) per hour increases. Inspections performed too fast will then result in decreased detection rate. However, we have not done any assessment of "optimal" inspection rates here. Also note, that the individual inspection rate is a *part* of the total inspection rates mentioned e.g. in Table 6.

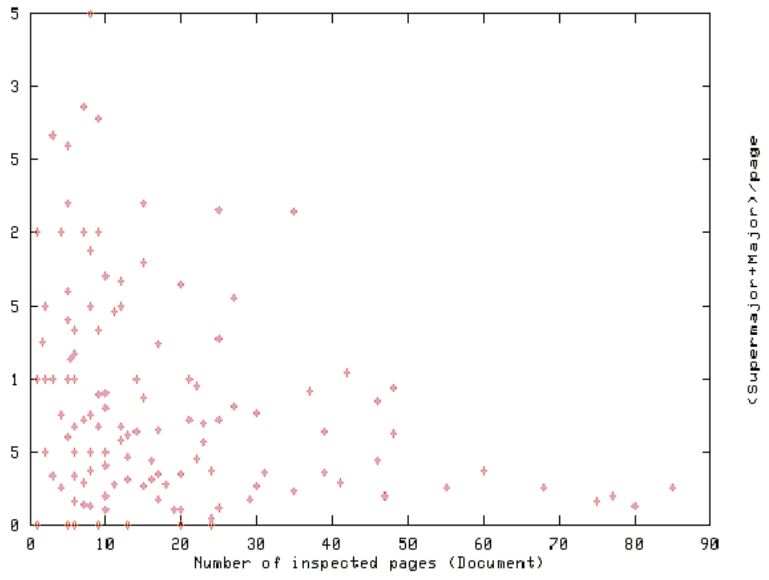


Figure 2. Number of pages inspected per hour and number of defects detected per page, Study 1.

5.3Q3: How cost-efficient are the inspection meetings?

Table 7 shows the effort consumption for each step of the inspections in Study 2, including defect fixing. Effort *before* individual reading and inspection meeting has been proportionally distributed on these two activities.

	Preparation	Inspection Meeting	Defect fixing	Sum
Person-hours [h]	5563	3215	11737	20515
[%]	27.12 %	15.67 %	57.21 %	100.00%

Table 7. Effort consumption for inspection and defect fixing, Study 2.

Note that 57.2% of the "inspection-time effort" is spent on defect fixing in Study 2 (11,737 of 20,515 person-hours), while only 21.1% are spent on such (311.2 out of 1473.7 person-hours) in Study 1.

Table 8 below for Study 2, shows the number of defects recorded in preparations, meetings, and total.

	Major defects		Super Major defects		Sum defects	Defect detection Effort	Cost-effectiveness
	[#]	[%]	[#]	[%]	[#]	[h]	[h:m per defect]
Inspection Reading	4356	97.2%	122	2.7%	4478	5563	01:15
Inspection Meeting	380	96.9%	12	3.1%	392	3215	08:12

Entire inspection	4736	97.2%	134	2.7%	4870	8778	01:48

Table 8. Cost-effectiveness and defect classification from inspections, Study 2.

As mentioned, the defects are split into two categories:

♦ **Major:** Defects that can have a major impact later, that might cause defects in the end products, and that will be expensive to clean up later.

♦ **Super Major:** Defects that have major impact on total cost of the project.

It turns out that 8% of defects found by inspections in Study 2 are found in meetings, with a cost-effectiveness of 8h:12min of person effort per defect. Compared to Function Test and System Test, inspection meetings are indeed cost-effective in defect removal.

5.4Q4: Are the same kind of defects found in initial inspection preparations and following inspection meetings?

We will also like to investigate what type of defects are found during preparations versus inspection meetings. Note, that we do not have data on whether inspection meetings can refute defects reported from individual inspections ("false positives"), cf. [Votta93]. Our data only report new defects from inspection meetings ("true negatives"). Table 8 from Study 2, shows that 2.7% of all defects from inspections are of type Super Major, while the rest are Major.

For preparation, the Super Major share is 2.7%. For meeting the share is 3.1%, i.e. only slightly higher. We therefore conclude that inspection meetings find the same "types" of defects as by individual preparation.

No such data were available in Study 1.

5.5H5: Relation between number of defects in Field Use and module complexity/modification-rate

H5: *There is a significant linear regression between number of defects found in Field Use and module complexity/modification-rate.*

Initial digression: Before we looked at the Field Use data in our database, we made plots for Design Inspections, Deck Check, Function Test etc. to visually check if there were any obvious (cor)relations between defects in any phase and module complexity/modification-rate – but all our plots showed just "noise".

Intuitively, we would say that the defects (failures) in Field Use might be related to the complexity and/or modification-rate of the module. The modification-rate indicates how much the module is changed from the base product, based on *source code* changes. Module complexity is represented by the number of states in a corresponding *design* document, taken from a state machine diagram (BDFC document) and reported by TeleLogic's SDL tool, SDT. For new modules the modification grade is zero. There were only N=7 data pairs for modules that had data through all the phases.

For **H5** the regression equation (one of many possible ones) can be expressed as:

$$N_{fu} = \alpha + \beta N_s + \lambda N_{mg}$$

where N_{fu} is number of defects (failures) in Field Use, N_s is number of states, N_{mg} is the modification rate (grade), and α, β, and λ are coefficients. **H5** will be accepted if β and λ are significantly different from zero, and if the significance level (p) for each of these two coefficients is lower than 0.10. The following values were estimated:

N_{fu}= -1.73 + 0.084*N_s + 0.097*N_{mg}

Predictor	**Coefficient**	**StDev**	**t**	**p**
Constant	(α) -1.732	1.067	-1.62	0.166
Complexity, N_s	(β) 0.084	0.035	2.38	0.063
Modification-rate, N_{mg}	(λ) 0.097	0.034	2.89	0.034

Here are s = 1.200, R^2 = 79.9%, and $R^2_{(adj)}$ = 71.9%, where s is the estimated standard deviation about the regression line, R^2 is the coefficient of determination, and $R^2_{(adj)}$ is similar but adjusted for degrees of freedom. That is, if a variable is added to an equation, R^2 will get larger, even if the added variable is of no real value. To compensate for this, $R^2_{(adj)}$ is chosen as coefficient of determination.

The values for estimated coefficients are given above, along with their standard deviations, *t*-value for testing if the coefficient is 0, and the *p*-value for this test. The analysis of variance is summarized below:

Source	**DF**	**SS**	**MS**	**F**	**p**
Regression	2	28.68	14.34	9.96	0.018
Error	5	7.20	1.44		
Total	7	35.88			

In the above table, DF is the degrees of freedom, SS is the total sum of squares corrected for the mean, MS is the mean of squares, F is a Fisher observer for the F-test, and p is the significance level for this test.

It should be noted that the constant (α) is not significant, but that the complexity and modification-rate are significant. The F-Fisher test is also significant. Therefore, the hypothesis **H5** *can be accepted* based on the results from the regression analysis.

5.6H6: Relation between number of defects in Unit Test and module complexity

H6: *There is a significant linear regression between number of defects found in Unit Test and module complexity.*

H6 uses Study 2 data shown in Table 9 for some modules (or rather a specific block in some modules, see below) coming from project A-E. The last Project F was excluded, because the relevant modules were only present in Projects A-E. For each block, a project has information on the number of defects and defect densities from inspections (these data are not used here, see **H7.5**). There is also information on module complexity (i.e. states as in **H5**) and number of defects found in Unit Test. Module modification-rate is not included, as in **H5**, due to missing data. Data slots marked with "-" mean that data are missing or no module (block) exists.

Data are collected for each document (block) type, and a module in each phase consists of several kinds of document types. One document type is therefore selected through all the phases/releases -- namely BDFC. However, because few modules are structurally stable over several releases and have relevant data, only 12 (!) modules out of 443 could be used for this analysis. Table 9 shows a total of N=18 complete data pairs, i.e. defects in Unit Test and module complexity, coming from two releases, namely Projects A-B. Project C had only two data points, and was excluded. All these 18 data points are treated together, thus no separate analysis for each project A and B. See, however, the digression in the end of the section.

The later **H7** hypothesis will investigate longitudinal relations within phases of a release and across releases, using the same data as in Table 9 below.

Module name	Project A			Project B			Project C			Project D			Project E		
	Defects/page	Complexity	Defects found in Unit Test	Defects/page	Complexity	Defects found in Unit Test	Defects/page	Complexity	Defects found in Unit Test	Defects/page	Complexity	Defects found in Unit Test	Defects/page	Complexity	Defects found in Unit Test
SUSAACA	0.04	72.0	25	0.28	80.5	3	-	-	-	-	-	-	-	-	-
SUSAACT	0.10	177.5	12	0.10	179.0	4	-	-	-	-	-	-	-	-	-
SUSCCTB	0.42	117.5	58	0.80	120.5	24	-	-	-	-	-	-	-	-	-
SUSCR	-	-	-	0.13	95.5	11	-	-	-	3.80	89.00	-	-	-	-
SUSCWC	0.29	-	23	0.10	-	-	-	-	-	-	-	-	-	-	-
SUSCWHF	-	-	11	0.50	-	-	-	-	-	-	-	-	-	-	-
SUSCWP	0.06	220.5	7	0.27	240.0	13	-	-	-	-	-	-	-	-	-
SUSSCR	0.08	244.5	22	-	295.5	34	-	-	-	-	-	-		-	-
SUSACF	0.14	47.0	32	0.37	62.5	28	-	-	-	-	-	-	0.24	66.0	-
SUSAP	0.26	67.0	42	-	-	10	-	-	-	-	-	-	0.04	78.0	-
SUSCCTA	0.34	269.5	118	-	297.5	132	1.00	299.5	3	-	-	-	-	-	-
SUSCS	0.06	257.0	14	0.90	267.5	34	0.18	254.5	21	-	-	-	-	-	-

Table 9. Defect density, complexity, and Unit Test defects for BDFC documents for different modules and projects (releases), Study 2.

The regression equation to state **H6** can be expressed as:

$U = \alpha + \beta C$, where U is #defects in Unit Test, C is module complexity, α and β are constants.

H6 will be accepted if β is significantly different from zero and its significance level is less than 0.10. The following α, β-values were estimated (see also Table 10 below):

$U = 26.5 + 0.039 * C$.

Predictor	*Estimate*	*Standard error*	*t*	*P*
α	26.49460	14.27865	1.86	0.0791
β	0.039072	0.109087	0.36	0.7242

Table 10. Estimated values, Study 2

This indicates that the linear regression line and accordingly **H6** *must be rejected.*

Digression: A correlation of 0.36 (again N=18) was computed between U and C, thus showing a low level of covariation. If the data set was limited to only Project A, the correlation was 0.15 (N=9). For Project B alone, the correlation was 0.55 (N=9). In other words, **H6** is a dead end.

5.7H7: Relation between defects across phases and releases for individual modules

*5.7.0.0.0.1***H7:** *There are significant correlations between number of defects in the same modules across phases and releases.*

The objective behind **H7** is to see whether the registered defects from Design Inspections can be used as a predictor for test defects in later phases or for any defects in later releases. I.e. will a defect-prone module (block) in Design Inspections continue to be so in the later phases and releases?

We have used data from Study 2 and 3, and have split **H7** into six sub-hypothesis **H7.1-H7.6**, as described in the next subsections. Tracing blocks *within a project* has only been attempted for project A (**H7.1-H7.4**), since this was the only project that contained enough registered defects for the same kind of blocks. Tracing blocks *across releases (projects)* has only been attempted for projects A and B (**H7.5-H7.6**), for similar reasons but using defect densities as an indicator. Other releases (projects) also contain information on blocks listed in projects A and B, but they lack comparable information on the same type of documents across phases and releases. For all hypotheses **H7.1-H7.6,** a *threshold correlation value of 0.8* will be used to decide if a given hypothesis can be accepted or not. Since we have rather few data observations, no significance level etc. was included. As an overall result, some hypotheses were accepted, but the majority were rejected. To perform a more satisfactory hypothesis investigation and corresponding data analysis, we need a more comprehensive data set on reported defects in several document types across phases and releases.

H7.1: Relation between number of defects in Design Inspections and Desk Check, Study 3

H7.1: *There is a significant correlation between the number of defects found in Design Inspections and Desk Check in Project A.*

Table 11 shows the raw data and the results. Each block is represented by a portfolio of four document types from the Design Inspections, namely BD, BDFC, PRI and SPL (see Table 1). The accumulated number of defects in the document types within one block (rightmost column) is then compared to the number of defects found in Desk Check (leftmost column). To see how the different documents contribute to the portfolio, the individual correlation per document was also calculated.

Project A	Desk Check	BD inspections			BDFC inspections			PRI inspections			SPL inspections			Sum selected block inspections		
Block	Defects	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P
SUSAACA	7	3	29	0.10	6	134	0.04	1	15	0.07	4	9	0.44	14	187	0.07
SUSAACT	1	3	27	0.11	7	68	0.10	1	20	0.05	2	11	0.18	13	126	0.10
SUSCCTA	82	25	23	1.09	40	116	0.34	3	10	0.30	9	3	3.00	77	152	0.51
SUSCCTB	41	18	22	0.82	45	106	0.42	6	8	0.75	5	2	2.50	74	138	0.54
SUSCS	10	11	38	0.29	6	107	0.06	8	8	1.00	3	7	0.43	28	160	0.18
SUSCWP	3	7	45	0.16	7	110	0.06	52	126	0.41	2	6	0.33	68	1287	0.24
Correlation Coefficients		**0.9473**			**0.8687**			**-0.3048**			**0.9696**			**0.6897**		

Table 11. Correlation between number of defects in Design Inspections and Desk Check, Study 3.

Table 11 shows an overall correlation coefficient (rightmost, bottom figure in table) of 0.69 (N=6) for this hypothesis. **H7.1** *must be therefore rejected.*

H7.2: Relation between number of defects in Design Inspections and Function Test, Study 3

H7.2: *There is a significant correlation between the number of defects found in Design Inspections and Function Test in Project A.*

H7.2 is similar to **H7.1**, but for Function Test not Desk Check. The block types BD, BDFC, and PRI are selected. Table 12 shows the data material and the results.

Project A	Function Test	BD inspections			BDFC inspections			PRI inspections			Sum selected block inspections		
Block	Defects	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P
SUSAACA	4	3	29	0.10	6	134	0.04	1	15	0.07	10	178	0.06
SUSAACT	2	3	27	0.11	7	68	0.10	1	20	0.05	11	115	0.10
SUSACF	6	9	28	0.32	8	57	0.14	0	4	0.00	17	89	0.19
SUSCCTA	10	25	23	1.09	40	116	0.34	3	10	0.30	68	149	0.46
SUSCCTB	10	18	22	0.82	45	106	0.42	6	8	0.75	69	136	0.51
SUSCS	3	11	38	0.29	6	107	0.06	8	8	1.00	25	153	0.16
Correlation Coefficients		**0.8767**			**0.9297**			**0.1648**			**0.9136**		

Table 12. Correlation between number of defects in Design Inspections and Function Test, Study 3.

According to Table 12, the correlation coefficient turned out to be 0.91 (N=6) for this hypothesis. Thus **H7.2** should *be accepted.*

H7.3: Relation between number of defects in Design Inspections and System Test, Study 3

H7.3: *There is a significant correlation between the number of defects found in Design Inspections and System Test in Project A.*

H7.3 is similar to **H7.1-H7.2**, but for System Test. The block types AI, BD, BDFC, PRI, and SPL are selected. Table 13 shows the data material and the results.

Project A	System Test	AI inspections			BD inspections			BDFC inspections			PRI inspections			SPL inspections			Sum selectedb lock inspect-ions
Block	Defects	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P	Defects
SUSAACA	1	4	14	0.29	3	29	0.10	6	134	0.04	1	15	0.07	4	9	0.44	18
SUSAACT	1	3	14	0.21	3	27	0.11	7	68	0.10	1	20	0.05	2	11	0.18	16
SUSCCTA	2	16	23	0.70	25	23	0.109	40	116	0.34	3	10	0.30	9	3	3.00	93
Correlation Coefficients		**0.9976**			**1.000**			**0.9997**			**1.0000**			**0.9974**			**0.999**

Table 13. Correlation between number of defects in Design Inspections and System Test, Study 3.

Table 13 shows a correlation coefficient of 0.999 (N=3) for this hypothesis. Thus **H7.3** *must be accepted,* although the data material is rather meagre.

H7.4: Relation between number of defects in Design Inspections and Field Use, Study 3

H7.4: *There is a significant correlation between the number of defects found in Design Inspections and Field Use in Project A.*

H7.4 is similar to **H7.1-H7.3**, but for Field Use. The block types BD, BDFC, and PRI are selected. Table 14 shows the data material and the results.

Project A	Field Use	BD inspections			BDFC inspections			PRI inspections			Sum selected block inspections		
Block	Defects	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P	Defects	Pages	D/P
SUSAACA	2	3	29	0.10	6	134	0.04	1	15	0.07	10	178	0.06
SUSAACT	1	3	27	0.11	7	68	0.10	1	20	0.05	11	115	0.10
SUSACF	3	9	28	0.32	8	57	0.14	0	4	0.00	17	89	0.19
SUSCCTA	6	25	23	1.09	40	116	0.34	3	10	0.30	68	149	0.46
SUSCCTB	6	18	22	0.82	45	106	0.42	6	8	0.75	69	136	0.51
SUSCS	7	11	38	0.29	6	107	0.06	8	8	1.00	25	153	0.16
Correlation Coefficients		**0.7760**			**0.5574**			**0.8546**			**0.7127**		

Table 14. Correlation between number of defects in Design Inspections and Field Use, Study 3.

Table 14 shows a correlation coefficient of 0.71 (N=6) for this hypothesis. Thus **H7.4** *should be rejected.*

H7.5: Relation between Inspection-level defect densities for BDFC modules over two consecutive releases – projects A and B, Study 2

H7.5: *There is a significant correlation between defect densities for BDFC (Block Data Flow Charts) modules across two consecutive releases -- Projects A and B.*

H7.5 uses the same data from Study 2 as **H6,** where also the defect densities are shown. Because of lack of comparable data, only Project A and Project B were used. However, the correlation turns out to be only 0.30 (N=7), so **H7.5** *must be rejected.* Digression*:* Even if the "outlier" module SUSCS is excluded, the correlation becomes only 0.66.

Digression. If we try an *augmented hypothesis* **H7.5'** to relate the number of *defects in Unit Test* across the same two projects (just to exploit the available data material from **H6**), we get a promising correlation of 0.86 (N=9, with six modules in overlap with **H7.5**). However, if we now exclude the suspicious SUSCS module, the correlation falls to only 0.30 – so we find no clear inter-project defect pattern for Unit Test either.

H7.6: Relation between Inspection-level defect densities for BD modules over two consecutive releases -- projects A and B, Study 3

H7.6: *There is a significant correlation between defect densities in Design Inspections for BD (Block Data) modules across two consecutive releases -- Projects A and B.*

Table 15 shows the relevant data and results.

Document type	Block Description (BD) inspections		
Block	Defects per page (defect density)		**Correlation Coefficient**
	Project A	Project B	
SUSAACA	0.1034	0.1000	**0.5729**
SUSAACT	0.1111	0.1000	
SUSCCTB	0.8182	1.7000	
SUSCWP	0.1556	0.1915	
SUSCACF	0.3214	0.6410	
SUSCS	0.2895	2.6000	

Table 15. Correlation for inspection-level defect densities for BD modules between projects A and B, Study 3.

Since the correlation coefficient was only 0.57 (N=6) for this hypothesis, we decided to test whether the suspicious block SUCS was making a difference. This proved correct and a recalculated correlation *without* the block SUCS turned out to be 0.999 (N=5)! However, it is improper to omit at "outlier" data value from such a small sample. Therefore **H7.6** *should also be rejected.*

All in all, **H7** *can neither be accepted (two supporting subhypotheses), nor rejected (four non-supporting subhypotheses).* As mentioned before, more studies are needed.

6.Conclusion

After analysis of the data, the following can be concluded for Ericsson in Oslo:

Software inspections are indeed cost-effective: They find about 70% of the recorded defects, take 6% to 9% of the development effort, and yield an estimated saving of 21% to 34%. I.e., finding and correcting defects before testing pays off, so indeed "quality is free".

7% of the defects from inspections (3% in Study 1, 8% in Study 2) are found during the final meeting, while 93% are found during the individual reading. Almost the same distribution of defects (Major, Super Major) are found in both cases. However, Gilb's insistence on finding many (serious) defects in the final inspection meeting is not supported here.

By comparison, [Votta93] reports that 8% of the defects are found in the final inspection meeting. Votta therefore proposes to eliminate them, since they are costly (7-14 times less cost-efficient than individual inspections in our studies) and since their logistics is bothersome (binding up many busy people and thus victims to sudden cancellations). However, inspection meetings are indeed cost-efficient compared to Function Tests (6 times more cost-effective in Study 1), and presumably to later tests too. Inspection meetings also fullfill important social functions, like dissemination of knowledge and promotion of team spirit. At Ericsson they also serve to give an overall quality check and approval of design documents.

Individual inspection reading and individual Code Reviews are the most cost-effective techniques to detect defects, while System Tests are the least cost-effective.

The recommended inspection rates are not really followed, since only 2/3 of the recommended time is being used.

The number of defects in Field Use for one concrete system (Study 1) was significantly related by linear regression with its module complexity (number of states) and modification-rate (**H5**). However, in we could not find a significant linear regression between the number of defects in Unit Test and module complexity in Study 2 (**H6**). Manual analysis of many of the data plots for defects in inspection/test and complexity in Study 1 revealed no obvious covariations. Thus the overall pattern in **H5** and **H6** is non-conclusive. When discussing all this with the Ericsson people, we learned that Ericsson often puts the best people to develop intricate modules, and that more attention is generally devoted to complex software. This may explain why few significant correlations were found. Anyhow, more studies are needed here.

Some of the hypotheses show that there is a significant correlation between defects found in different phases within a project. That is, if a block is defect-prone in Design Inspections, it will continue to be so in later test phases and releases. However, we had insufficient data to establish whether defect-prone modules "carried on" in a predictable pattern (**H7**).

The collected, defect data has only been partly analyzed by Ericsson, so there is a potential for further analysis also by themselves. However, NTNU has already “data mined” the material quite exhaustively.

The defect classification (Major and Super Major) is too coarse for causal analysis in order to reduce or prevent future defects, i.e. a process change, as recommended by Gilb. We also lack more precise data from Function Test, System Test, and Field Use.

It is somewhat unclear what these findings will mean for process improvement at Ericsson. At least they show that their inspections are cost-effective, although they could be tuned wrt. recommended inspection rate (number of inspected pages per person-hour, as part of overall inspection rates). However, all such tuning requires a stable process, see below.

On the other hand, more fine-grained data are necessary for further analysis, e.g. for Root-Cause-Analysis (also recommended by Gilb). Such defect classification seems cheap to implement at defect recording time, but is almost impossible to add later. However, the database tool for defects will need to be expanded with more data fields, so this is not a minor process change after all. The reported process has also become less relevant after these studies had been performed, since Ericsson is increasingly adopting UML and Java as a system platform, instead of SDL and PLEX. However, on this new platform no alternative, formal inspection procedures have been defined and put in use. We are now negotiating with Ericsson to try out novel object-oriented inspection techniques [Travassos99].

Inspired by these findings, NTNU wants to continue its cooperation with Ericsson on defect studies in the context of mutual projects. Their defect database seems under-used, so these studies may encourage a more active utilization of collected data.

Acknowledgements: We wish to thank colleagues at Ericsson for their time and interest in these investigations, and the Norwegian SPIQ project on Software Process Improvement for economic and contextual support. We also thank Oliver Laitenberger from Fraunhofer IESE, Kaiserslautern for insightful comments.

7.References

[Adams84]) Edward Adams: "Optimizing Preventive Service of Software Products", IBM Journal of Research and Development, (1):2--14, 1984.

[Basili96] Victor R. Basili, Scott Green, Oliver Laitenberger, Filippo Lanubile, Forrest Shull, L. Sivert Sørumgård, and Marvin V. Zelkowitz: "The Empirical Investigation of Perspective-Based Reading", Empirical Software Engineering, Vol. 1, No. 2, 1996, pp. 133-164.

[Dybaa00] Tore Dybå, editor: *SPIQ metodebok for prosessforbedring i programvareutvikling – v3.0* (in Norwegian), SINTEF/NTNU/UiO, Trondheim and Oslo, Norway, Jan. 2000, IDI-rapport no. 2-2000, ISSN 0802-6394, ca. 200 p.

[Fagan76] Michael E. Fagan: "Design and Code Inspection to Reduce Errors in Program Development", IBM Systems J. Vol 15, No. 3, 1976.

[Fagan86] Michael E. Fagan: "Advances in Software Inspections", IEEE Trans. on Software Engineering, SE-12(7):744--751, July 1986.

[Gilb93] Tom Gilb and Dorothy Graham: "Software Inspections", Addison-Wesley, London, UK, 1993.

[Hantho99] Øivind Hantho: "An Empirical Study of Inspection and Testing Data", IDI NTNU, Trondheim, Norway, 19 Jul. 1999. 147 p., EPOS TR 360 (diploma thesis).

[Lott96] Christopher M. Lott and Hans Dieter Rombach: "Repeatable software engineering experiments for comparing defect-detection techniques", Journal of Empirical Software Engineering, 1(3), 1996. 34 p.

[Marjara97] Amarjit Singh Marjara: "An Empirical Study of Inspection and Testing Data", IDI. NTNU, Trondheim, Norway, 22 Dec. 1997. 108 p., EPOS TR 308 (diploma thesis).

[Skåtevik99] Børge Skåtevik: "An Empirical Study of Historical Inspection and Testing Data at Ericsson" (forthcoming), IDI, NTNU, Trondheim, Norway, 8 Feb. 1999. 90 p., EPOS TR 350 (diploma thesis).

[Travassos99] Guilherme H. Travassos and Forrest Shull and Michael Fredericks and Victor R. Basili:
"Detecting Defects in Object Oriented Designs: Using Reading Techniques to Increase Software Quality", Proc. Conf. on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'99). In ACM SIGPLAN Notices, Vol. 34, No. 10, Oct. 1999, p. 47-56", Denver, Colorado, 3-5 Nov. 1999.

[Votta93] Lawrence G. Votta: "Does Every Inspection Need a Meeting?" In Proc. ACM SIGSOFT 93 Symposium on Foundation of Software Engineering. ACM Press, December 1993.