Reviews and Inspections - An Approach to the Improvement of Telecom Software Development Process

Stjepan Golubić
etkgos@etk.ericsson.se

ETK/D/N
Ericsson Nikola Tesla d.d.
Zagreb, CROATIA, 10000

Dean Marušić
etkmdn@etk.ericsson.se

ETK/D/S
Ericsson Nikola Tesla d.d.
Split, CROATIA, 21000

Abstract

The role of reviews and inspections in telecom software development process is analyzed. The necessity for combination of reviews and inspection is explained. The relation between inspections and testing is analyzed. Some results of analysis of collected inspections and testing data are given. The need for an intelligent inspection tool is presented and a conceptual proposal is given.

1. Introduction

Errors are inherent to any aspect of human activities. The development of software product is not an exemption. As a purely intellectual product, it is among the most labor-intensive, complex, and error-prone technologies in human history [1]. According to the IEEE –STD-610, *software product is the complete set, or any of the individual items of the set, of computer programs, procedures, and associated documentation and data designated for delivery to a customer.* In general, the software product is a document or code. The injection of defects, or faults, during the development of software product cannot be avoided. A *defect* is a problem with the product that is detected sometime during its development. On the other hand a *failure* occurs when a software product (executable code) does not perform the task the user requires and expects of it during the operation [2]. A *failure* in human readable documents may cause misunderstanding or an incorrect action [3]. Software product development team should keep in mind the *Murphy's law* all the time. Telecom systems have a long period of exploitation. Software needs have to be organized so that different activities, such as design, production, operation and maintenance, can be facilitated.

2. Specifics of telecom systems

The most functions of the telecom systems are implemented by means of software. Software accounts for 70-80% of the development costs of a switching system [4]. There are more and more services that are provided by the telecommunications network, and more and more users that invoke these services. The total software volume continues to grow, as systems become increasingly complex. One consequence of the transition from traditional switching ("telephone exchanges") to a network concept is that we must be able to describe the network as our "machine", designed to give the end –user the very services he wants. Mobile telephony, Integrated Services Digital Network (ISDN), and Intelligent networks (IN) are examples of processor capacity-consuming services. Computers have a wide range of uses in different areas of life, but the requirements are not the same for all of them. Telecom systems are the real-time systems with the following properties:

- interaction with the external world,
- handling of multiple unrelated inputs,
- concurrent processing,
- speed and precision in response, and
- a complex behavior.

The computer in a telecom system must produce correct responses within definite time interval. Even more, jobs are not carried out in the same order as they occur, but rather in order of priority. The requirements on the usability, availability, quality and reliability of such systems are very stringent. It is expected that they have an excellent In-service performance (ISP) with the zero down time and the minimal maintenance cost [5]. Telecom software has to be continuously upgraded because of a long period of exploitation. To satisfy the customer needs, the software packages have to be specified, produced, and delivered rapidly and correctly. In recent years ISP, including system down time, has become a very important sales argument. The improvements in the area of telecom software development process have become a critical issue. Because of that, the development of telecom software must be a carefully planned, controlled, methodical, predictable and measurable process [6, 7].

3. The role of Reviews & Inspections process

There are several factors necessary for successful business operations:

- the establishment of the software development process is naturally the basic one,
- the technical in-depth knowledge, or competence, which is necessary to build-up the product, and
- finally the most problematical one is the skill to develop the high quality product at a low in-house cost in continuously compressed time-to-market window. In other words, an organization should have good people doing the right thing in an effective way.

Traditional business operations are normally organized as fixed, functional-oriented, line units on one side, or temporary project organization on the other side. Today, the process management appears as a necessary complement to the line and project management. The process management is well known and widely used approach to improve quality and productivity in a systematic way. The emphasis is on the continuous improvement [8]. The difference between a successful and unsuccessful company lies in the rate of improvement. Every our business process starts with a customer and ends with a customer. Both, the line organization and the project organization, employ the processes to carry out their assignments. Business operation improvements shall be managed cross-functionally, process oriented, in order to focus on the customer's satisfaction, originating from the customer's needs. Software development is the process of developing software from customer requirements (**Figure 1**).

Improvements of only engineering, or technical, processes such as requirement analysis, design and coding are not enough. It is not realistic to expect that defect-free product could be ever developed, because of the Murphy's law. It is necessary to have a cost–effective way for the defect detection and early elimination. The place and role of Reviews and Inspections (R&I) process in achieving that goal is analyzed here. **Figure 2** shows the R&I process overview with the main group of activities (sub-processes).

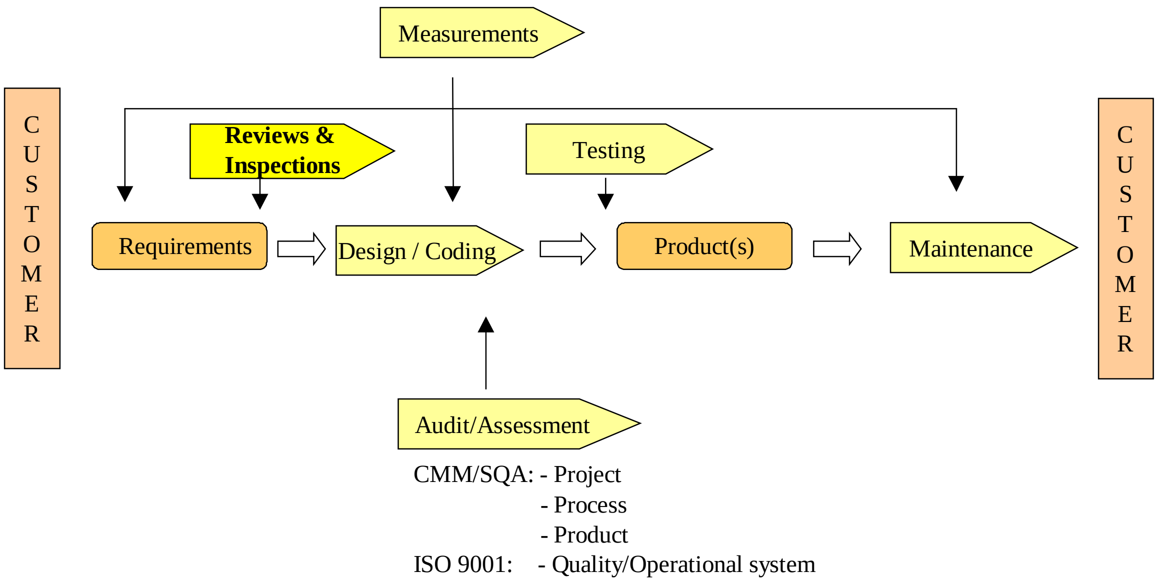

Figure 1. The software development and maintenance flow

3.1. Combinations of reviews and inspection

The IEEE Standard 1028-1988 (Software Reviews and Audits book) defines a review as *an evaluation of software element(s) or project status to ascertain discrepancies from planned results and to recommend improvement*. There are a number of different review methods in practice, for example, management reviews, technical reviews, frequent reviews, one third presentation, walkthroughs, and so on. Although the IEEE Standard has defined some of the review methods, there is still no common terminology of this area in practice. Very often the same names have different meanings in different organizations, and sometimes different names are used for very similar review methods. Inspection can be defined as the most formal review, which results in a logged defect data and a written report (the inspection record). According to the ANSI/IEEE Std. 729-1983, an inspection is defined as *a formal evaluation technique in which software requirements, design, or code are examined in detail by a person or group other than the author to detect faults, violations of development standards, and other problems.*

One of the main telecom designers' complaints to the implementations of such "classical" inspection was that it doesn't deal enough with the technical correctness of the design. Because of that reviews, with the focus on the technical aspects of the software product, are added to solve this problem. An example of such combination of reviews and inspection performed on a document is presented in

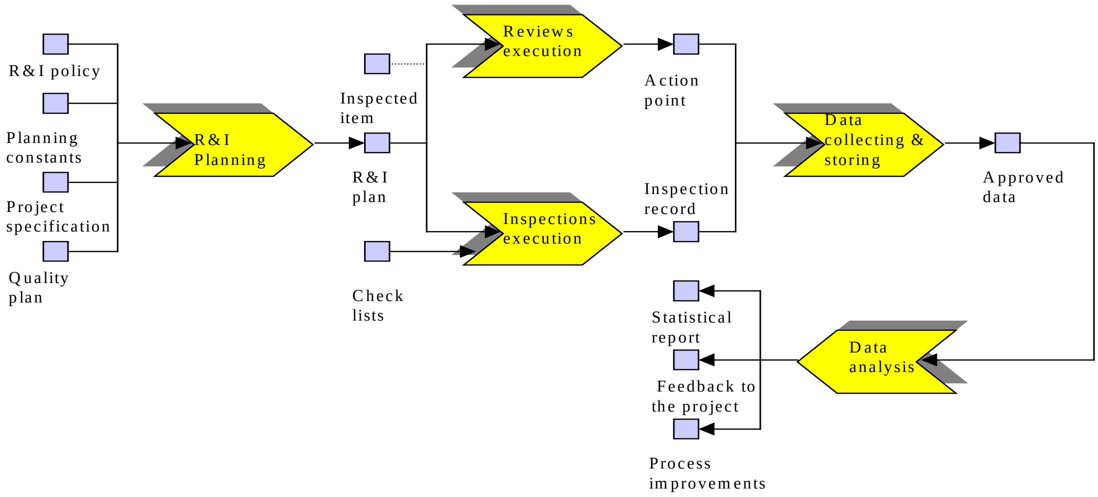

Figure 2. The Review & Inspection process overview

Figure 3. The type and the number of reviews can be different for different types of documents, but an inspection is always done in the end. Anyhow, reviews are not so formal as inspections. The result of performed review requires some improvements work defined through action point. The number of action points per document are registered and included in the final inspection record.

The purpose of such combination of reviews and inspections is not only to detect and correct the defect as early as possible, but also to train and teach design team members, especially new and young designers, in both reviews and inspection process and design process as well.

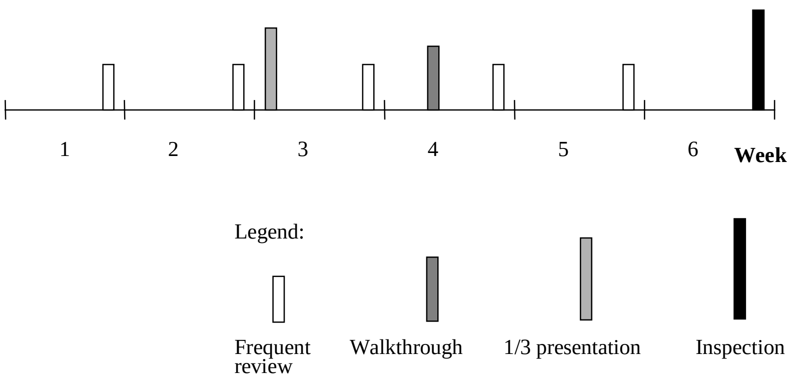

Figure 3. An example of reviews and inspection combination

3.2. Relation between inspection and testing

The establishment of the "defect-free" software development process is not possible without inspections and testing. Both activities aim at evaluating and improving the quality of the software deliverable product before it reaches the customer, but they can't replace each other. It is visible in the **Figure 1** that inspections are mostly front-end activities and testing back-end activities. Both belong to the third level of Capability Maturity Model (CMM) hierarchy and could be treated as quality control activities as shown in **Table 1**. Today it is normal to see testing as an integral part of design. Software product has to be designed in such a way that enables its testability.

The same principle should be valid for higher levels of design, for document preparations (starting from the interpretation of customer requirements). Documents have to be prepared in such a way that enables easier inspections. They should be "inspectable" which means that document, or source code, should be readable, well structured, modular, commented, cross-referenced, traceable, an so on. The main problem with inspections is that decision about defect and document correctness strongly depends on subjective and psychological human factors.

The executable code is the place where inspections are replaced by testing. While inspections examine static documents, or source code, using only human brain, testing dynamically evaluates working product, an object or executable code, by using interpreter, emulator, simulator and/or test plant, depending on the product test phase. Anyhow, planning of testing starts in very early design phases, and inspections are also used for improvements of the testing documents. The efficiency value of code inspection may allow comparing of inspection and testing efficiency. The efficiency value of document inspection is an almost unique value because inspections are practically an only method for the verification of documents. To be comparable with defect found in executable code, the *major defects*, found by inspections in document, should be defined as those defects that could result in defect in executable code, and are potential sources of failures in working product.

The R&I are one of the key factors in achieving a high quality product because they are implemented in the development process from the very beginning. It is an economical mechanism for continuous building-in of quality into the product. The deliverable software products and R&I process itself should be audited by SQA (Software Quality Assurance) function. The complete operational system is certified according to ISO 9001 standard.

Table 1. R&I position in CMM hierarchy

CMM level	Processes/Activities	Operational level	Product quality level (predicted fault density*)
5	Root Cause Analysis Continuous improvements Defect prevention	Total Quality Management	.00X
4	Quantitative quality goals Measurements Data Analysis Defect prediction	Quality management	.0X
3	Inspection (Peer review) Testing Defect detection	Quality control	.X
2	Reviews, Audits, Assessments, Non-conformities Corrective actions	Quality assurance	X (single digit)
1	Ad hoc activities	No quality activities	XX (double digit)

Note: *) Reference 2

4. An example of analysis of data collected from the software development projects

Data has been collected manually and analyzed according to the R&I process flow shown in **Figure 2.** The inspection data has been collected during documents design phases and testing data have been collected during function test, system test and the six months operation period. The results for a few projects are presented in **Figure 4**. The cumulative data of all projects for two years period is presented in **Figure 5**. The data analysis has shown that the increased use of inspection has resulted in decreasing of the product fault density and increasing of the project lead time precision. The achieved product quality is in accordance with the predicted fault density for the CMM level 3 (**Table 1**).

been detected during document or code inspections. This is a strong indication for further improvements.
Also, one of the main conclusions is that it is not enough, any more, just to collect data from projects and use it as historical date base in the next project. The complexity of today's systems and diversity of projects requires in-project decisions that should be based on the data collected in earlier project phases. The exit criteria from the inspection should not be based on the number of detected faults, but the prediction of the number of those remaining after the

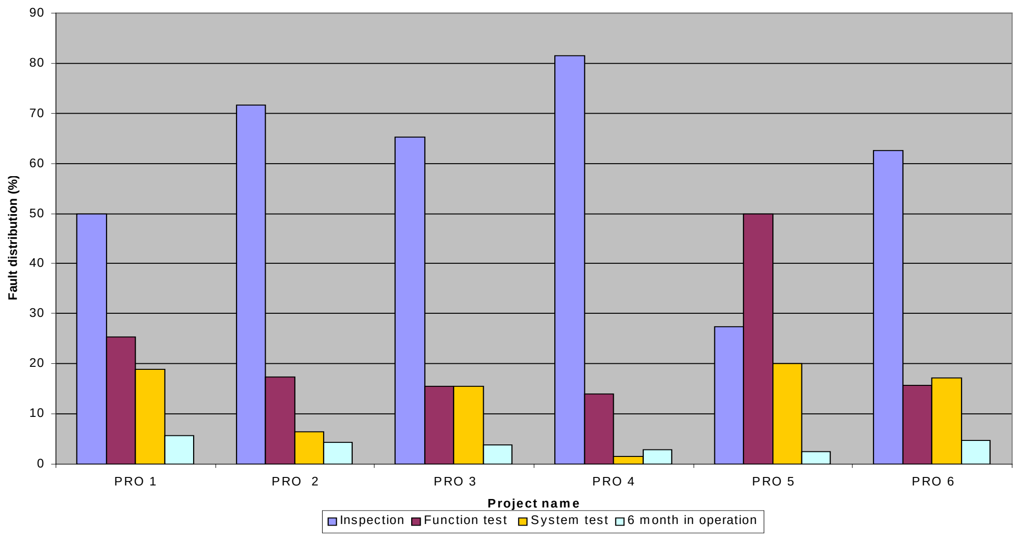

Figure 4. The fault distribution per project

From the graphs of fault distribution in projects appears that when less faults are found during inspections, higher faults are correspondingly found in function test or system test. Data analysis per quarters has also shown that product quality increases when more inspections are carried out.
Using collected data in this way it was possible to evaluate if the complete development process is under control. The estimation of the number of faults, which could be expected in the similar future projects, is also possible. It is also a rough indication of the area where some improvement is needed. But there are also a lot of economical benefits. It is well known and generally accepted that the cost of detecting and correcting defects rises exponentially with each next development phase. Simple calculation in our case has shown that the cost ratio between fault correction in inspection and in operation is approximately 1 to 20. The performed root cause analysis of faults detected during testing phases had showed that a lot of faults should have

inspection is carried out. Even more, such analysis should be used as help in prediction of faults in coming testing phases. As with any measurement, the collected data must be consistent and accurate. However, this does not mean the data has to be too precise. It is more important to understand the relationships among them and to be in the right order of magnitude.
Accurate data requires that some primary data elements are defined and commonly understood and accepted, especially the definition of major defect, size of inspected item and time for inspections. Even after these elements are defined, people will still need to make value judgements based on recorded data. These judgements are not likely to produce consistent and useful conclusion until there is a common understanding of how the data will be analyzed and used. The results of data analysis have to be used for the improvements both the product and the process itself as well. For achieving these goals, an effective support tool is a must.

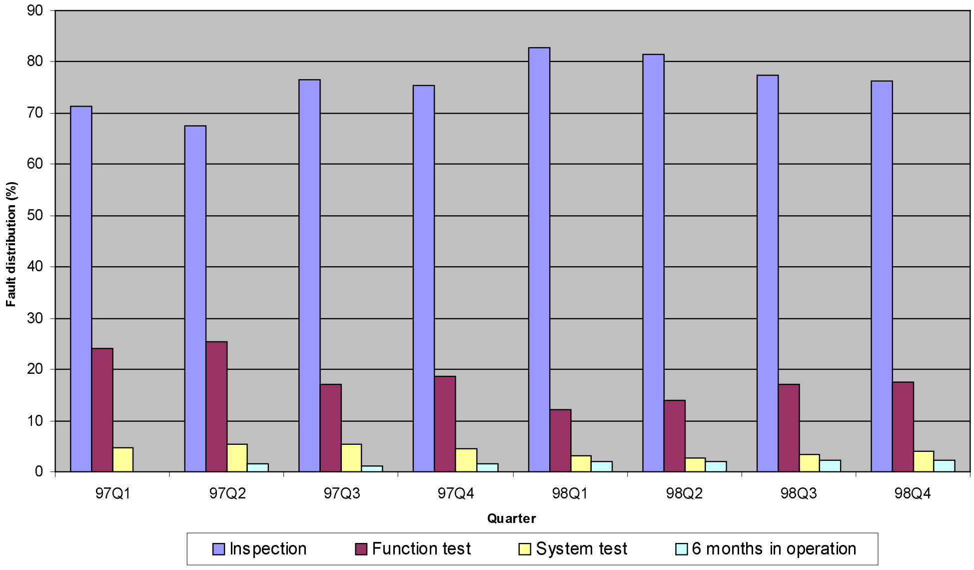

Figure 5. Fault distribution – cumulative data

5. The concept of an intelligent inspection tool

The concept of an intelligent tool, aimed to enable and help in line, project and process management, is shown in **Figure 6**. Proposed concept of an intelligent inspection tool is built up on a base of three different subsystems, or three different tools. The idea for proposed concept was necessity to use some of newer knowledge acquisition techniques, based on Artificial Intelligence (AI) models, in order to improve managing of inspections and the whole telecom software development process [9,10]. Inspection knowledge engineering driven approach is represented by the proposed inspection tool concept.

Subsystem A has function of Decision Support System (DSS), with requirements to provide on-line form for fill-in inspection data, assure collecting of inspection data into inspection database, analyze inspection data and present analysis results in different report forms.

In proposed concept, inspection database has to be maintained by Subsystem A. Throughout the telecom software development process large volumes of inspection data are collected. Amount of data is increasing continuously during the time. Typically, these data are used to provide endless "facts and figures" about inspection results in organization in different report forms. But, inside inspection data there is a number of hidden trends, patterns and relationships between number of inspection attributes which are not visible in a case that inspection data is processed manually, or by OLAP (On Line Analytical Processing) which use linear analytical models. Solution for automated discovery of previously unknown patterns and automated prediction of trends and behaviors is implementation of 'Data mining' subsystem, based on processing by non-linear analytical models. Inspection database, established and 'fill-in' through Subsystem A, is input for 'Data mining' subsystem (Subsystem B).

Subsystem B is based on application of AI techniques (neural networks, genetic algorithms, and rule induction) to discover hidden trends patterns and relationships to large quantities of inspection data stored in inspection database by usage of Subsystem A functionality.

Subsystem B is a subsystem with 'data mining' functionality, which provide possibility to predict future trends and behaviors of inspections, allowing business to make proactive, knowledge driven decisions inside inspection process. Data mining scour inspection databases for hidden patterns, finding predictive information that inspection expert may miss because it lies outside their expectations. Inspection process management is user of such subsystem, with responsibility to analyze and design knowledge discovered by subsystem in to input for Subsystem C.

Subsystem C is in function of Knowledge Base Decision Support (KB-DS) built on the base of inspection expert knowledge and inspection knowledge discovered by 'Data mining' subsystem (Subsystem B).

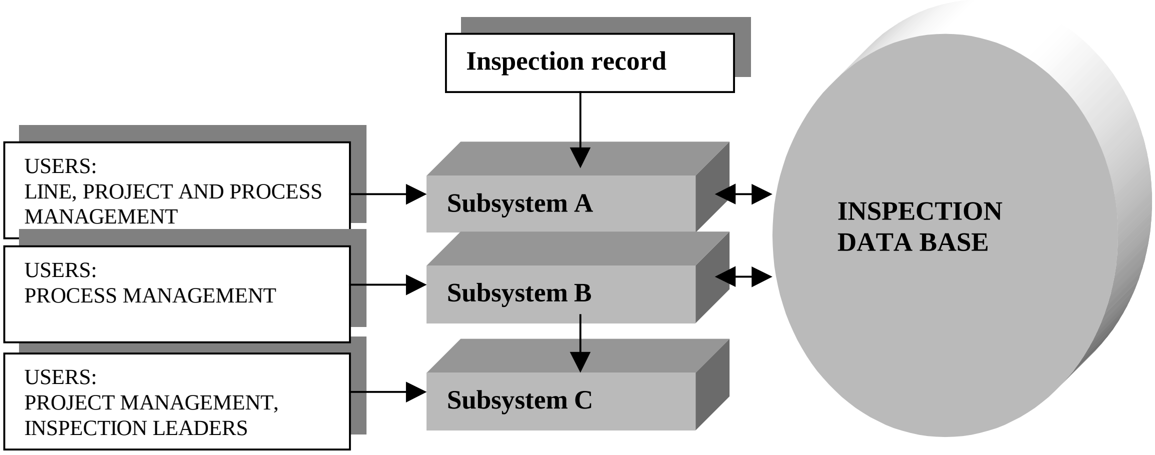

Figure 6. A concept of an intelligent inspection tool

Inspection KB-DSS is used for planning of inspections and for inspections execution based on the inspection knowledge and organization's historical inspection data.

Proposal is to use Decision tree logic or Rule base logic to capture inspection expertise for planning and inspection execution. Proposed subsystem provides spread of inspection parameters defined by line and process management (inspection exit criteria, optimum checking rate per document type, number of inspectors, and so on) through whole organization. Subsystem C can be divided in two main parts for decision making inside inspection process: inspection execution and inspection planning.

Inspection execution part has to provide exit criteria for inspection of different document types as a support to inspection moderator decision making. Inspection planning is performed in early project phases, so inspection planner can not handle precise data for inspection planning. At the beginning of a project, real value of changes inside document or number of pages of whole document for inspection is unknown, only rough estimation can be provided to inspection planner. Because of that, implementation of *fuzzy logic*, one of artificial intelligence techniques, is necessary inside subsystem C. Inspection planning (regarding document size) can be correctly performed by "fuzzification", or modeling of human imprecise reasoning using fuzzy sets. Fuzzy set for "document size" can be explained in terms *small*, *medium* and *high*, with their fuzzy membership functions.

Inspection knowledge covered by the subsystem should be maintained and controlled by Inspection process management.

Aim of implementation of the proposed inspection tool is to drive the inspections inside organization by usage of data mining and advanced analysis techniques to help identify risks, to retain customer and to target prospect.

Efficient design and implementations of proposed inspection application imply the use of already developed commercial development shells. There is a number of compatible development environments for design of every specific DSS, Data Mining and KB-DSS inspection applications.

Integration of proposed inspection subsystems into one inspection system provides a base for successful managing of inspection planning and inspection execution. It also enables continuous improvements in inspection area by usage of inspection expert knowledge and knowledge about inspections in organization discovered by AI applications implemented inside the system.

6. Conclusion

Faults affecting products when used by customer should normally be detected and corrected before they reach the customer. Even more, they should be predicted during the project. Improving the software development process improves the quality of software products and the overall performance of the software development organization.

The role of inspection in software development process is not only directed to detect and correct faults as early as possible. It is a method for measuring and controlling the whole development process, and the way to predict and prevent faults in later phases.

Of course, reviews and inspections can't assure software quality by themselves. To better assess the effect of inspections on the product and the development process, inspection data is most useful when combined with other processes and product measurements, especially with testing. Inspections should be planned together with testing. One common verification strategy has to be implemented in the project taking into consideration complete development flow. The third element that should be combined with inspections is a product and process audit. All together can significantly improve and assure the quality of software in cost effective way. It also significantly reduces the number of failures in operation. This is very important for both customer satisfaction with the product functionality and for reducing maintenance cost. For further improvements of reviews and inspections an intelligent tool concept is given.

The majority of work, which results are partially presented in this paper, was done during last two years. We hope these efforts have contributed to ETK/D in acquiring the ISO 9001 Certificate, and reaching the CMM Level 3. At the corporation level, those activities have been a part of the Ericsson Systems Software Initiative program, and Good Practice activities.

References

[1] H. Krasner, “Using the Cost of Quality Approach for Software”, *CROSSTALK, The Journal of Defense Software Engineering*, pp. 6-11, November 1998.

[2] R. G. Ebenau, S.H. Strauss, *Software Inspection Process*, McGraw-Hill, Inc., 1993.

[3] T. Gilb, D. Graham, *Software Inspection*, Addison-Wesley, 1993.

[4] *Understanding Telecommunications 1*, Ericsson Telecom, Telia and Studentlitteratur 1997.

[5] *Software Reliability Handbook*, Ericsson Telecom AB, 1995.

[6] J.P. Paulish, A.D. Carleton, Case Studies of Software–Process–Improvement Measurement, *COMPUTER*, pp.50-57, September 1994.

[7] S. Golubić, “The analogy between the software and hardware development and production methodologies”, *Conference on telecommunications ConTel 95*, ITA 14 (1995) 1-3, pp.141-149, 1995.

[8] *The Capability Maturity Model*, Guidelines for Improving the Software Process, Carnegie Mellon University, Software Engineering Institute, Addison-Wesley, 1997.

[9] R. K. Michel, L. B. Methlie, *Knowledge-based Decision Support Systems with Applications in Business*, John Wiley & sons, Second edition, 1995.

[10] D. Marušić, “Decision Support System for Software Inspection Process”, *Seminar work*, Faculty of Organization and Informatic, Varaždin, 1999. (in Croatian