ESSI Process Improvement Experiment (PIE)

Improvement of Process Architecture Through Configuration & Change Management and Enhanced Test Strategies for a Knowledge-Based Test Path Generator

Project 24,078 - IMPACTS2

Final Report

Version 1.2

22-March-2000

DTK GESELLSCHAFT FÜR TECHNISCHE KOMMUNIKATION MBH
Palmaille 82 • D-22767 Hamburg • Tel. 040.389970.0

Purpose

The Final Report is a mandatory executive report detailing the experience and the lessons learned during the experiment, from a technical and business point of view. It is meant for public dissemination and should not contain company confidential information.

Document Code

none

Availability

Public

Version History

Version 0.1, 01.07.98, Sven Nordhoff
Initial document

Version 1.0, 30.08.98, Harry Debler
Revised

Version 1.1, 19.12.99, Harry Debler
Revised, BOOTSTRAP Self-Assessment results added

Version 1.2, 22.03.00, Harry Debler
Revised according to Reviewers comments

Contents

Figures

Tables

1 Executive Summary

This report describes the results of the ESSI Project Nº24,078, IMPACTS2. The project provides an important contribution to the establishment of test process- and configuration & change management procedures.

The approach has been applied to and validated in a baseline project in which we are using a development process that maintains the continuity in all stages of the software life cycle. In this manner the procedures for testing and the powerful and effective introduction of configuration & change management is necessary. For this reason we noticed that the processes in these areas should be improved. The improvements are measured with the help of software process metrics. The main result of the experiment is that the continuity of the development methodology with an early establishment of test procedures is important to provide a high-quality software system with reduced efforts

The PIE has been submitted as a direct consequence of a BOOTSTRAP-compliant Self Assessment [8] which ranked the software processing unit at 1.25. Due to the lack of a formal quality system the PIE has the task but also the ability to establishes new methods and procedures to improve these processes.

Special attention was given to make the approach in conformity with national and international standards because the baseline project is a safety critical application. For this purpose, it was important that the procedures and methods that were exemplary established in this project, fulfilled all necessary standards and guidelines (e.g. CENELEC EN50128 [9]).

The results are especially valuable for companies, which develop safety critical systems. For companies interested in the aspects of this project, it was established a site in the internet which provides results and experiences[1].

This project was carried out in the framework of the Process Improvement Experiment (PIE) from the European System & Software Initiative (ESSI) with a financial contribution by the Commission of the European Communities.

2 Background Information

2.1 Objectives

The objective of the PIE project was to define and apply efficient and productive processes for testing and configuration & change management. For the testing processes this approach includes the definition and implementation of test methods and procedures and for the configuration & change management process it was the establishment of configuration identification, control and accounting procedures. For each process the selection of tools and their integration into the development environment was necessary. The main emphasis of the project furthermore was the definition of several quality aspects which are included in the company wide quality system. To do this it was necessary to pay attention to the company wide standards for safety critical applications and environments.
This approach is applied and validated on a baseline project. This underlying project is concerned with the development of one of the strategic products of our company, namely the test path generator for analogous relay-based circuits, for use in the railway environment. This is a knowledge-based system, known as Relay-MASTER and part of the software product

[1] http://www.dtkhh.de/impacts2/

family of the DTK (X-MASTER). The software for this system is developed in C++ and Allegro Common LISP using the object-oriented paradigm. High transparency, high traceability up to the developer in the case of incidents and very high product quality is required. It will be demonstrated how the process architecture for this test path generator can be improved by applying configuration & change management methods as well as enhanced test strategies. The definition and evaluation of suitable metrics, that valued the improvements of the different processes, resulted in an clearly improved product quality, concerning reliability, maintainability, testability and safety.

2.2 Involved companies and their role

This PIE is performed by the **DTK GESELLSCHAFT FÜR TECHNISCHE KOMMUNIKATION MBH**. The DTK is a company engaged in software quality assurance, safety-evaluation and safety-realisation. DTK is acting as a consultant, and as a software development and system house. Thus software development plays a significant, business-relevant role. In this context one of the strategic goals of the company is the permanent raise of the quality of its products and services, by continuously improving the maturity of its processes. The introduction of modern techniques and tools is an essential step on this way. Further organizational, methodological and technological steps are planned beyond this PIE, for 1998 (e.g. concerning the company-wide introduction of object-oriented analysis and architectural design methods) and 1999 (e.g. defining a company-wide standard for the development of process metrics).

The DTK got assistance from the consultant subcontractor **KODA GmbH**. This company assisted in the definition of suitable metrics and support in the development and execution of training procedures regarding these metrics.

2.3 Starting scenario

As a part of the introduction of a quality system, the software development unit of DTK has undergone a BOOTSTRAP-compliant self-assessment in December 1995. The whole unit and two projects were considered during the assessment in order to evaluate the current software development practices, to identify potential areas of improvement as well as to define an action plan in relation to the overall objectives of DTK. The result of the self-assessment was a maturity level of 1.25 for the software development department. This value is in line with the average maturity level of the European industry, and reflects the difficulties that small companies face in establishing a comprehensive suitable quality system. Such result essentially means that although there are some elements of repeatability across the existing practices, most of such practices are highly person dependent and the level of standardisation across projects is virtually non-existing.

Besides several strengths within the analysed software producing unit and the examined projects, the most significant weaknesses have been identified as follows:

- Lack of metrics and process measurements in general
- Lack of company-wide standards for testing
- Lack of procedures for configuration & change management
- Missing of a comprehensive quality system

Based on the results of the assessment, DTK decided to introduce as the baseline building block of the companies quality system a suitable international standard on software development. By now the V-Model standard and its four sub-models as well as the ISO/IEC 12207 (IT-SW Life cycle processes) have been considered. An implementation of the V-Model

- even in a tailored version - seems to be unnecessarily complex and expensive, when looking at the DTK-specific requirements and needs for project development. Thus preference is currently given to the ISO/IEC 12207.

The actions had been concentrated on the above mentioned points 2 and 3. These actions were the main topics of the process improvement experiment. From a business point of view the company was willing and in a position to implement methods for testing and configuration & change management. The technical staff of the DTK were very interested and motivated in performing and transforming the processes into action. Also the baseline project was motivated to join the effort. The baseline project and the PIE were staffed with experienced engineers on the one hand and graduates that have come directly from university into the company on the other hand.

2.4 Working Plan

The following table is summarising the work plan for the experiment, including key phases, milestones, planned efforts, and project deliverables, as given in the Project Program (Section 3.9) of the PIE. Descriptions of the phases, milestones, and deliverables are given in section 3.5.

Workpackage reference	Start date	End date	Effort (person-days)	Deliverable reference
WP1: Project Management	**1.3.97**	**2.6.98**	**60**	**PPR1, PPR2, PPR3, MTR, FR, CS**
WP2 / PH I: Initialization	**1.3.97**	**19.3.97**	**25**	**DPP**
WP3 / PH II: Workbench Selection	**19.3.97**	**7.5.97**	**25**	**TSR**
Selection of C&C-Management. and Test Tools				
WP4 / PH III: Application 1	**7.5.97**	**14.11.97**	**112**	**PMR, CSK, TPK, IAR**
WP5 / PH IV: Application 2	**17.11.97**	**6.5.98**	**355**	**CSS, TPS, EDG, EQM, FAR**
WP6 / PH V: Result Evaluation	**7.5.98**	**3.6.98**	**7**	**FR, BSR**
BOOTSTRAP-Compliant Self-Assessment				
Dissemination Activities	**4.8.97**	**2.6.98**	**52**	
Total			**636**	

Table 1 Workpackage references

2.5 Expected outcomes

The main aspect of the PIE project was to increase the software quality of the delivered product. The survey of the BOOTSTRAP-compliant self-assessment results in potential areas of process improvements. Therefore we expected that the development process should be significantly improved by focusing on testing and configuration & change management.

The experiment should be validated on that parts of the baseline project, which components were already multiple reused (e.g. parts of the upper mentioned rule base). The benefit for the whole project as well as for other projects will be most significant for the successful evaluation of appropriate techniques and tools.

The PIE project was supposed to deliver concepts, practical procedures and integrated tools for the test and configuration & change management processes. Therefore, the configuration & change management system will manage all of the documents and software in the project (sources, data, executables, tools, documents, and so on). The interrelationships and

interdependencies between the objects administrated should be transparent and well-documented. Process steps should be automated to a certain extent. They become thus well-defined and replicable. In order to evaluate the potential success of this approach to process improvement the following features are considered: [24]

- Implementation of a configuration management plan
- Version Control
- Workspace and Release Management
- Build Management
- Change Management

In the area of testing, the experiment should bring the processes to a level where we have well-defined test procedures and methods for the main test tasks. It hopefully will come out, which test methods are useful for different test tasks and how these methods can be combined to an effective test environment. The knowledge about testing technologies in the company will be enhanced and afterwards introduced in other projects, internally as well as externally (consulting). We will mainly concentrate on the following aspects [25]:

- Definition of Systematic and Automated Test Procedures
- Development of Reproducible Test Procedures
- Definition of Standardised Test Documentation

The involved staff will be trained in configuration & change management and test methods and tools.

We intend to define a set of process metrics, which should give us the ability to get more precise information about process relationships and about an improvement potential in our development process [4].

3 Work Performed

3.1 Organisation

The project has been managed by a project manager, who allocated approximately 50 % of his time for the process improvement experiment. The quality manager of the DTK was also involved in the PIE. His task was to support the process improvement steps and to evaluate the benefits of the performed actions for the baseline project. In the different phases a changing number of developing engineers, testing engineers and persons who have experiences in quality assurance are involved.

Phase I of the experiment is done by the project manager, the quality manager and one software engineer who had the responsibility to introduce the involved staff members to the project concerning scope and goals. For this the interface to the underlying baseline project (in this phase a subset defined as the **B**aseline **P**roject **K**ernel (BPK)) have been established.

In Phase II and III besides the project manager and the quality manager, two additional software developers are working 50% of their total work time for establishing and translating the procedures and methods from the PIE to the baseline project. They have performed their normal developing work, while evaluating new tools for configuration & change management and testing using the new methods which are defined during the experiment.

In Phase IV 2 additional project members were involved in the baseline project. In this phase a new subset of the baseline project (**b**aseline **p**roject **s**ubset) was established. Training task and enhancement of the quality standard were established.

In phase V the results and experiences that had been made during the experiment were evaluated. A final BOOTSTRAP-compliant self assessment had been carried out in order to better evaluate and value the results of the project.

3.2 Technical environment

The technical environment in the company involved consists of Windows NT workstations as development platforms. Furthermore UNIX-based Servers and Workstations and a MicroVAX with the operation system VMS are used within the company. The target platform for the experiment was Windows NT.

The following table shows the systems and tools that were used to support the introduction of test processes and configuration & change management:

System	Tools	Comments
Windows NT-Server 4.0 SP3		
Server operation system	ClearCase® License Server	
	ClearCase® VOBs and VIEWs	data storage for versions and releases
	McCabe License Server	
Windows NT-Workstation 4.0 SP3		
Workstation operation System	ClearCase® System	Configuration Management Tool
	ClearCase® Snapshot Views	local data storage for ClearCase
	McCabe ToolsSet® System	Static Test System
	Developer Studio 97	C++ Development System
	Cantata++®	Dynamic Test System
	Purify NT®	
	Visual Test®	Capture Replay Tool
	CCWord	ClearCase Word Integration
	CCRose	ClearCase Rational Rose Integration
SUN Solaris 2.4/2.6 UNIX		
	Allegro Common Lisp 4.3	LISP development system

ClearCase, Cantata++, McCabe ToolSet, RATIONAL Rose C++, Purify NT and Visual Test were introduced during the PIE project (see Tools Selection Report[16])

3.3 Training

Two training events were held during the experiment. First, during phase III, for all key project members an introduction and overview of the existing testing and configuration management techniques was performed. Later in phase IV, all other project members were introduced in the processes and methods and were guided in using the different tools. Workshops and training courses about the test and configuration management tools were given by the vendors of the introduced products. Although the area of object-oriented

analysis and design (e.g. requirement analysis and specification, design for testing) was not covered during the experiment, the need to train the involved project members in testing of object-oriented systems was necessary. For this purpose a workshop was held that was introducing to the topics design for testing, design-by-contract and requirement testing.

3.4 Role of the consultant

The consultant KoDa GmbH has established the metrics procedure. During this phase a lot of meetings were necessary to introduce the project members in the methods of GQM analysis. Furthermore one representative of the company has lead our 1'st in-house seminar.

3.5 Phases of the experiment

The 5 major workpackages of the PIE were carried out in consecutive phases. A sixth workpackage was containing the overall project management. A seventh workpackage is related to the dissemination activities performed in parallel to the phases of the experiment. The contribution of the subcontractor is mentioned where appropriate.

This chapter describes the content of different phases and the associated workpackages and their experiment actions, reporting activities, dissemination actions and milestones.

3.5.1 Project Management

This phase comprised the overall project management and control of the PIE, in terms of time, costs and quality. The project management has been performed according to DTK's standard procedures. This includes monthly reporting by the project staff to the project management and monthly reviews of the project status based thereon, done by the project management and the company's quality manager. The project management particularly has been concerned with risk avoidance to ensure the successful outcome of the experiment.

In the course of this phase the **Periodic Progress Report** (PPR1, PPR2, PPR3) has been prepared as well as the **Mid-Term Report** (MTR), the **Final Report** (FR, this report), and the final **Cost Statement** (CS).

3.5.2 Phase I: Initialisation of the experiment

During initialisation all the involved staff members have got an introduction to the PIE project concerning scope and goals. Especially, the quality manager of the company has attended the meeting. All organisational issues have been settled, comprising the definition of the project team as well as the determination of the roles each member will play and the task assignment for the project staff. The exact scope of the underlying baseline project and the interface definition between the PIE and the baseline project were defined. Furthermore, parts of the baseline project the PIE has been applied to, were determined. One part was defined as the „baseline project kernel" (BPK). This forms the input for the in-depth tool and process evaluation during Phase III (Application 1). The selection was based on the following criteria:

- multiple reused under different configurations, thus having major significance for the successful evaluation of C&C-Management and test techniques and tools
- high module or object complexity in order to stress-test selected tools and techniques
- familiarity of key project members in order to avoid preliminary preparation overhead

As a result, this phase obtained a detailed workplan of the project.

3.5.3 Phase II: Workbench Selection

During this phase appropriate software tools for configuration & change management and testing have been evaluated and selected. This has resulted in a tool selection report which also documents the tool evaluation criteria. As far as available, experience from former ESSI / PIEs has been taken into account.

The specific features for the configuration & change-management have been closely related to the task of handling the different project documents produced within DTK's software engineering process. For this requirement a lot of tools exists. We decided to use **ClearCase**® for Windows NT from RATIONAL Corp., because this tool has the best integration in our development environment.

The selection of appropriate test tools was much more complicated than for C&C management. Especially for module and integration test tasks we found out that there is not only one tool available, that satisfies all our requirements for the envisaged test activities. Every test tool that we evaluated has its specific strengths and weaknesses. This perception was very important for the process of tool selection.

After an intensive evaluation we made the decision, that the test tools McCabe Toolset and CANTATA++ in combination are covering most of our requirements.

Furthermore **Purify NT**®, a memory leak detection program, and **Visual Test**®, a capture replay tool, satisfied our other necessities.

3.5.4 Phase III: Application 1

In this phase different tasks were established to introduce new process methods and procedures. The tasks are:

1. Evaluation of necessary methods and procedures

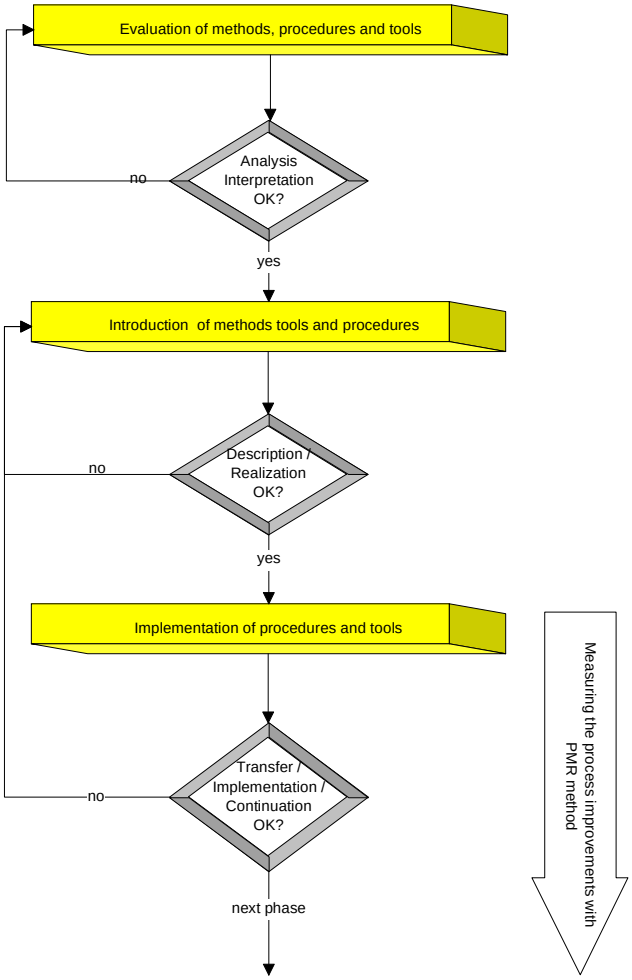
   - Analysing what to do (C&C, Test)
   - Analysing what methods and procedures other companies or organisations use (C&C, Test)
   - Decision about methods and procedures that we should use (C&C, Test)

2. Introduction of methods and procedures

   - Establishment of configuration management processes (description and flow-graphs)
   - Establishment of test management processes (description and flow-graphs)

3. Implementation of methods and procedures

   - Implementation of process procedures (C&C, Test)
   - Establishment of tools (C&C, Test)
   - Training of key project members



Tab. 1 Process graph for Application 1

Furthermore a valuation of the introduced process methods and procedures has been established. For this purpose, basic metrics have been developed and defined with assistance from the subcontractor, in order to measure the improvement of the processes. These results represented valuation criteria for the next phases, introducing further and detailed process improvements. The definition of the measurement process and of suitable metrics for our processes has been documented in the process metrics report (PMR). In the Appendix of this report we introduce in the main valuation tasks for our experiment.

The tool specific training has been performed by RATIONAL (ClearCase) and McCabe (McCabe ToolSet). For all other tools we used the accompanying tutorial material from the vendors of the tools.

As a result of the introduction of tools, methods and procedures in the underlying baseline project, an **I**ntermediate **A**pplication **R**eport (IAR) was generated. This report documents the progress in the introduction of tools and the required effort for implementing the experiment's tasks. In the middle of the experiment a lot of remarks of the baseline project were produced:

- introducing of new methods is cost and time intensive
- increasing effectiveness will be seen only in later phases of the project
- the first project working with new techniques will take longer and will be more expensive
- the knowledge about modern software techniques will increase the quality and productivity of the development staff

3.5.5 Phase IV: Application 2

In this phase process methods and procedures introduced in phase III were examined. Another subset of the underlying baseline project, the **b**aseline **p**roject **s**ubset (BPS), was established. In this phase the following tasks have been established:

1. Introduction of software production guidelines

   - Evaluation of criteria
   - Searching for standards and scientific studies about development guidelines
   - Introduction of development guidelines
   - Implementation of development guidelines within the quality assurance process of the DTK
   - Training of project members

2. Enlarge the quality manual of the DTK

   - Evaluation of constructive and analytic quality assurance methods
   - Searching for standards and scientific studies
   - Completion of the company's quality manual
   - Training of project members

3. Measurement of the process improvements

   - Reporting and Documentation of the measurement process of the experiment
   - Detecting problems in the process definition
   - Resolve problems and define new or changed process descriptions

4. Enlarge or correct the process definitions, methods, procedures or tool implementation and repeat previous phases of the experiment

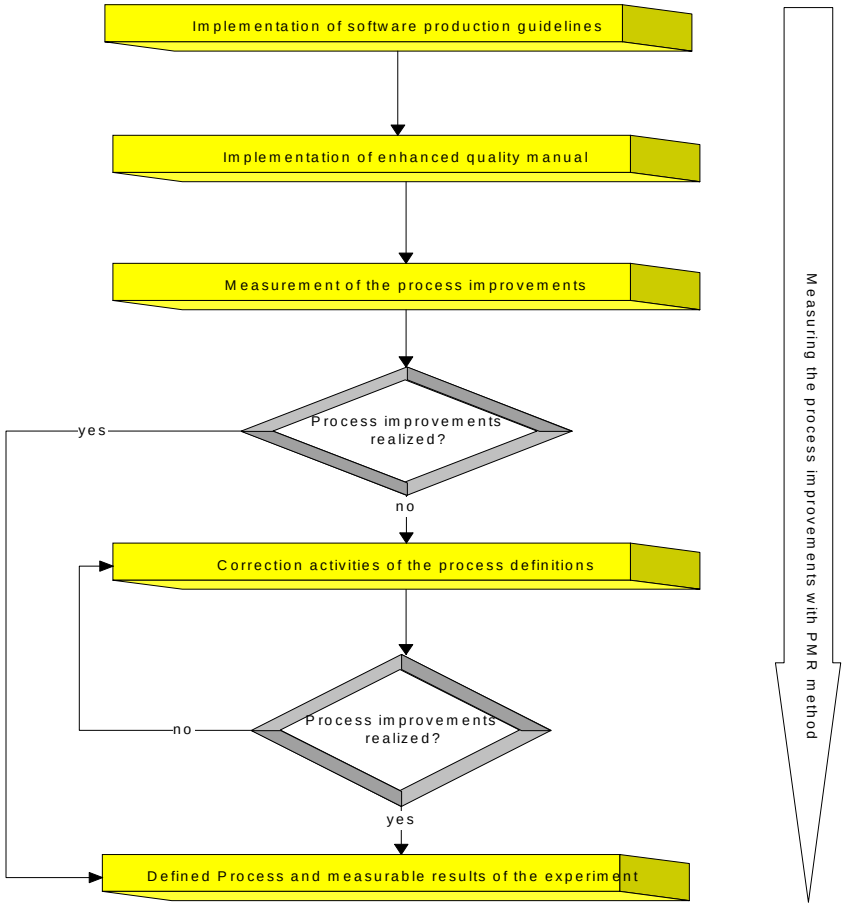
   - Implementation of new or changed process procedures
   - Change tool implementation

The software production guidelines and the company's quality manual will be enhanced according to the results achieved. The quality manager of the company will be strongly engaged in this task.

To measure the process improvements, we established in the end of the previous phase a measurement point to evaluate the resulting improvements. In the Annex of this report we will introduce in the main valuation areas of our experiment and distinguish and value the different measurement points and methods of the experiment.

Furthermore training and continued introduction of tools, techniques, and metrics to further members of the baseline subset project have been proceeded. 2 additional project members have been involved here.

The training has been performed in-house by already trained key project members and the subcontractor. The trainability of the methods have been examined as it is of major importance for wide-spread use of the methods and replicability of the PIE's results.



Tab. 2 Process Graph of Application 2

As a result of the introduction of tools, methods and procedures in the underlying baseline project, a **F**inal **A**pplication **R**eport (FAR) was generated. This report documented the progress, the introduction and behaviour of tools and the required effort implementing the experiment's tasks.

3.5.6 Phase V - Result Evaluation

In addition to the in-house evaluation of the methods and procedures examined during the experiment a final BOOTSTRAP-compliant self-assessment has been carried out in order to better evaluate the results of the project. The self-assessment results have been documented in an assessment report. The results and the process improvements will be shown in the appendix of this report.

3.5.7 Dissemination Activities

The results and experiences gained during the experiment have been and will be internally and externally disseminated and presented at official dissemination actions and through papers and presentations in other field specific conferences. Internationally, the mid-term results of the PIE have been presented in the ISCN'97 conference in Budapest/Hungary and the final results on the EuroSTAR 99 in Barcelona/Spain.

The main interests for our company are the railway environment and the area of quality assurance. During a 2-day workshop which was held in march 99 at DTK with international participation, the experiences gained from the PIE were introduced to some important companies which are busy with safety-critical software for railway control systems. The following companies were involved in the workshop: OSS/Denmark, Union Switch & Signal/USA, TUEV Rheinland/Germany, Arthur D. Little/England. The focus of the workshop was on the improvements made in the area of testing. The new European Standard for software in railway applications (prEN 50128, [9]) created by the CENELEC committee demands extensive testing on different levels to receive approval of a concerned system. The activities performed in the course of the PIE, especially in Unit and Integration Testing are very useful for companies which are involved in developing high safety-critical software for railway systems. DTK demonstrated their approaches for the different test tasks. This included the presentation of the used test methods and tools, the created procedures and of course the overall test management of such a project. The CM issues were also presented, where the main emphasise was on the CM treatment of work products generated during test processes. During the workshop there was an active discussion about reasonable test proceeding and how to optimise the work in order to reduce the necessary effort. We got also good new ideas how to improve the System Test processes. For companies in the sector of safety-critical systems there is a big necessity to improve their proceeding in System Testing and to provide the evidence of sufficient testing of the System Requirements.

3.6 Internal dissemination

For dissemination we have considered two In-house seminars, the first has been performed in August 97 and should reach the development and consulting staff of the DTK. This meeting was held as a working meeting to represent and examine the introduced methods and to have a response from the employees that weren't involved in the experiment. The second seminar was held in March 98 and has been directed to the management describing the main effects and advantages for the development and quality processes within the DTK. Both activities resulted in very positive feedback.

4 Results and Analysis

4.1 Technical

From a technical point of view in the PIE project IMPACTS2 we have achieved the following major results:

- Introduction of improved test processes involving:
  - Definition of characteristics and validity areas for different activities (e.g. module test, integration test).
  - Definition of general testing goals and quality levels.
  - Definition of quality aspects for developing software and referencing of quality aspects on test objects vs target objects (objects in the target system).
  - Introduction of a test team.
  - Introduction of responsibilities for the different test activities.
  - Definition of overall test level requirements (e.g. complexity and critically requirements).
  - Definition of criteria and requirements for the different test levels.
  - Training of the employees and increased knowledge and experiences in SW testing.
  - Introduction of test metrics.
- Introduction of improved configuration & change management processes involving:
  - Definition of the organisational structure of the process
  - Definition of general responsibilities
  - Establishment of a configuration control board (CCB)
  - Definition of configuration baselines
  - Definition of schedules and procedures for reviews and audits
  - Definition of the sub-process " configuration identification"
  - Definition of the sub-process " configuration control"
  - Definition of the sub-process " configuration status accounting"
  - Definition of the sub-process " audits and reviews"
- Metrics supported measurement of process improvements:

  A large number of metrics have been used to support different testing and configuration management analysis. The collection of several data sheets and figures in the annex shows an unambiguous improvement effect in the different processes. Positive effects on quality, productivity, efficiency, predictability and error prognoses and resolution have been shown in the baseline project. Different steps of this measurement process have been established. These are shown in the **P**rocess **M**etric **R**eport (PMR) [4]. This report describes the whole measurement process. For the task of metrics evaluation we had to define data extraction methods to gain the appropriate process data. A detailed measurement plan was established which describes all necessary measurement actions. Moreover data collection forms and a catalogue with valuation criteria have been created.
- Introduction of test and configuration management tools

  Different tools (see chapter 3.2) have been introduced to support the several activities defined within the test and configuration management processes. These tools have been integrated in the main development environment DEVELOPERS STUDIO 97® from Microsoft Corp.

4.2 Business

The results achieved in the PIE have been valued by the management of DTK. It is recognised that improvement in testing and configuration management is a very important step on the way to a high maturity software development organisation. As a matter of fact the activities of the PIE brought much benefit for the whole company as well as for other projects. It is just now common guideline for all new projects to use configuration management and to plan further test actions in the early project life time. Therefore the experiences of IMPACTS2 are used as much as possible. Also in existing projects it is investigated, whether the derived concepts can be taken over from the PIE project in an economical manner.

Moreover the performed process improvement experiment has lead to the understanding, that additional improvement actions in the scope of software development and quality management would be very useful for the company in the near future. Some other outcomes of the PIE are:

- Achievement of an ISO 9001 certification
- The gained experience and the results of the PIE have significantly improved the existing quality system. Now the step to achieve an ISO 9001 certification is much smaller than it was before.
- Achievement of compliance with CENELEC standards for railway applications

  This is one of the main topics of the company. Many projects of DTK are in the railway environment. The definition and installation of verification and validation processes is an example for typical activities in this area. Therefor the knowledge about the specific standards and the ability to adapt their contents to special applications is important. The experiences of the PIE project give a good guidance for this kind of work.
- Training in state of the art test and configuration management techniques
- A better understanding of costs and efforts needed to introduce new techniques

4.3 Organisation

The IMPACTS2 project did not lead to direct organisational changes. But a lot of additional responsibilities were assigned to individual persons in the company. One requirement to achieve the several tasks for configuration management was to assign one person as a configuration management authority (CMA). This person has the task to plan and overview the configuration management actions involving the communication of projects. This person spent about 30% of his working time for this task.

4.4 Culture

The IMPACTS2 project helped the management to recognise the following important topics:

- Experiences helped the management to recognise the importance of testing and configuration management.

- Testing is not limited to "product testing" at the end of the development. It is also useful to implement additional verification steps in different phases of the development. In this area the advantages of the V-Model had a big impact on the process definitions [1].
- The object-oriented paradigm has no better testing behaviour, but with support of the continuity for all stages of the software life cycle (OOA – OOD) the testing ability is better to handle. The experiment has shown that for inheritance and polymorphism the effort for testing is increasing.

- A well designed configuration & change management process leads to a better handling and transparency.

- The definition and usage of metrics is clearly necessary to identify and solve problems in process definition and gives a method to value source and design documents.

- Clarifies the expectations about strengths and weaknesses of tool-support

4.5 Skills

The most important skill improvement for the project members, was to learn the theoretical background of testing and configuration management and to practice specific abilities in software engineering. For the DTK the ability to learn about standards for the different processes in general and for the railway environment in specific (CENELEC [9]) was clearly necessary. For this, the project members have gained experiences in practically applying concepts described in standard or scientific studies.

Furthermore, it was important to value the project results. In this area, the improved skills estimating project efforts and quality aspects with the support of metrics had a great impact on the baseline project, as well as for other projects.

One of the most important experience is the extensive understanding of the features of current testing and configuration management tools. With this knowledge it will be much easier for us to select and introduce such tools in an appropriate manner in current and future projects. We have also realised that there is no test tool available on the market that fulfils all requirements of a test intensive project.

5 Key Lessons

5.1 Technological Point of View

Among the different lessons learned in the experiment and in the underlying project, the following ones have the greatest technical impact for our company:

- It was important for us to recognise that not all components of a software system have the same degree of criticality and consequently should not be tested all with the same intensity. Metrics can support in the selection of software components which are critical according to defined criterias. These criterias should reflect on the parts of the software which are high complex, unstructured, insufficient documented, etc. This leads to a test process which defines different stages of test intensities. This proceeding is especially helpful to reduce the effort for module testing. In the annex you can see the module testing process which has been developed in the duration of the PIE.

- To reduce the development and verification effort of whole subsystems, it is useful to divide a complete software system into subsystems with different safety requirements. The particular subsystems could be developed and tested to the extent required by the safety level. This process is supported by national and international standards, e.g. the CENELEC 50xxx norms, which provides so called 'Software Integrity Levels (SIL's)'.

- There is a big difference in the test requirements for functional oriented software on one side and object oriented software on the other side especially for the low level test tasks like integration and module testing. For functional based programs it is important to have the focus in determining complexity values, test case generation for dynamic testing and coverage measurements on the flow graph and/or the declared data. The testing of object oriented software will reflect more on the classes and the methods.

- The usage of special tools for testing and configuration management tasks leads to more effectiveness in these areas. But on the other hand it should not be overseen that the effort for introduction as well as for maintenance of those tools cannot be neglected. Only for the maintenance of the configuration management the responsible person spend about 20% of his working time and this in our relatively small baseline project. The introduction of a new colleague into test tools and methods should also not be underestimated. Depending on the experiences of the person it will take several weeks before he can use a test tool effectively.

- It is very difficult to get appropriate information for the definition of processes for testing and configuration management tasks. The contents of scientific studies on these items are often theoretical by nature and the existing standards have a very generic character, because they are designed to be adaptable to a great range of different kinds of projects. Due to this situation it is always a complicated task to create suitable processes.

- A configuration management process should be well defined and regarding all life cycle phases. The general ideas of configuration management seems to be easy to understand, but in detail there are some tricky relationships which have to be taken into consideration. An insufficient adapted configuration management will lead to certain problems which can be eliminated very hardly afterwards. The modern configuration management tools, e.g. ClearCase with it's special mechanisms like hyperlinks and labels offers the possibility to define appropriate processes.

5.2 Business Point Of View

The commitment of the management is an essential precondition for successful process improvement activities. This is influencing the acceptance of the new practices. The management of DTK was intensively involved in the PIE activities. It is recognised that the improvement of existing processes as well as the introduction of new processes will lead to positive mid and long term effects, if they are adequately adapted.

The main interest of the management is the dissemination and introduction of the new testing and configuration management processes into existing and new projects of the company.

As a matter of fact it is a problem for small and medium companies to investigate in scientific studies. Due to time and resource constraints this kind of work will often be disregarded. This performed PIE gave us the chance to research in different approaches for testing and configuration management.

The definition of suitable metrics to value e.g. the quality, the complexity and the testability of a product has been identified by the management as a plain necessity assessing product

relevant process. Also important for the management is the definition of process metrics to track the effort, the effectiveness or productivity of defined process methods. A good area for process metrics is the predictability of efforts in different development stages.

5.3 Strength and weaknesses of the experiment

There is a common positive feedback from the involved staff from the PIE project as well as from the baseline project. Nobody has regretted that this experiment has been performed. Nevertheless there are also weaknesses in the execution of the PIE. The following list gives a short overview of strengths and weaknesses:

+ The training effect on new technologies is significantly high. The knowledge on software engineering and software quality technologies has increased, especially in the area of testing and configuration management.

+ The transfer of the gained results and experiences of the experiment to other projects was forced especially through the in-house seminars and the training actions. The results have also influenced the company-wide quality system.

+ The 2 processes considered in the PIE are essential for DTK, but there are other processes which also need improvement. The performed actions have had a trigger effect for the establishment of further improvement activities. Therefore the selection of testing and configuration management as sample processes were a good decision.

+ There is a big need for consulting services in the area of testing and configuration management at our customers and the request for such services has also increased in the duration of the PIE. After the execution of the experiment, we can perform these consulting services better than we could have before.

- The main focus of the PIE in the performed testing activities was on module and integration test concepts. There is also a need for functional based test purposes on higher levels especially system testing and for the management of system test cases. These tasks should have given more attention.

- The testing of object oriented programs requires usually other procedures and methods. The approach for the PIE didn't took this aspect into account. The BPS has been a pure object oriented program, therefore we spent much more time to adapt our processes.

- We underestimated the effort for the introduction and maintenance of the configuration management process.

6 Conclusion and Future Actions

The following conclusion are very important not only for the development crew, but also for the management and the whole company:

- It is important to perform process improvement actions as real projects. This proceeding includes the creation of a project plan with all necessary items like project description, process definitions, time schedule, milestones, effort estimation, responsibilities, reporting, etc. Otherwise there is a high probability that the activities will not be executed seriously.

- Testing and configuration management should be planned in detail at the early beginning of a new project. The effort should not be underestimated.

- It is not enough to test a software product only at the end of a project on system level. Special types of errors can be detected easier and cheaper in early life cycle phases. Therefore testing should take place on different stages, e.g. on module, integration and subsystem level.

- There is no test tool available which is suitable for all kinds of testing and projects. Every tool has its specific strengths and weaknesses. It should be analysed in detail which testing requirements a project has and which tools are appropriate to fulfil these requirements.

- Not all parts of a software should be tested with the same intensity. Especially in safety critical systems it is pre-eminent to divide the whole system into subsystems with different safety requirements. These subsystems could be classified to different safety integrity levels according to safety standards, e.g. the new CENELEC 50xxx standards. Moreover it is appropriate to establish a graduated test process. Therefore the software under test has to be analysed to detect the critical components concerning quality aspects like degree of complexity, size and maintainability. Due to the criticality the separate components should be tested with more or less intensity or for extremely quality critical functions, a re-design should be forced.

- The responsibility for the quality assurance activities in a project should be assigned to a QA person in the project. It is not enough to have a company-wide quality manager. This proceeding will be mandatory for all existing and future projects.

- Process improvement actions leads to positive mid-term and long-term effects. There are less short-term effects after the introduction of new processes. The management has got more sensibility on this fact and has noticed that improvement actions will lead to success, if they are performed in an appropriate manner.

The following improvement steps are envisaged for the near future:

- The knowledge about system testing should be summarised and the belonging processes should be revised and extended. This testing stage was not the main interest of the PIE but it is as important as module and integration testing.

- The derivation of testing object oriented programs to a proper "object oriented testing" process should be on closer examination.

- The quality system of the DTK should be enlarged with respect to the definition and introduction of reviews and code inspections.

7 Glossary

BPS	Baseline project experiment subset. The overall subset of project modules handled in Phase IV of the experiment.
BPK	Baseline project kernel. Those modules of the baseline project singled-out for intensive evaluation of methods and tools in Phase III of the PIE.
BSR	BOOTSTRAP-compliant Self-assessment report
CM	Configuration management.
CR	Change request
CSK	Configuration Structure of the BPK
DTK	DTK GESELLSCHAFT FÜR TECHNISCHE KOMMUNIKATION MBH, main contracting organisation of the PIE
DWP	Detailed work plan
EDG	Software development guidelines
ESSI	
FAR	Final application evaluation report of the baseline project
GQM	"Goal-Question-Metricate", measurement technique for process improvement results, devised in the context of the ESPRIT project Application of Metrics in Industry (AMI)
IAR	Intermediate application evaluation report of the baseline project
IEEE	Institute of Electrical and Electronics Engineers, Inc.
OO	Object-oriented
OOA/OOD	Object-oriented analysis and object-oriented design. Software analysis and design methodology based on the object-oriented paradigm.
PMP	Process Metric Report
PIE	
PIR	Process Improvement Report
PP	Project Program
PM	Project Manager
TPK	Testing Process for the BPK
UML	Unified Modelling Language, new OOA/OOD modelling language created as unification from Booch's method, Rumbaugh's OMT, and Jacobsson's *Use-Cases*

8 Reference

[1] H. Balzert: Lehrbuch der Software-Technik, Software Management, Software Qualitätsicherung, Unternehmensmodellierung, Spektrum Akademischer Verlag, 1998

[2] Basili / Caldiera / Rombach: The Goal Question Approach, 1994

[3] Basili / Briand / Walcélio / Melo: A Validation of object-oriented design metrics as quality indicators, 1995

[4] Begemann, M; Nordhoff, S: Process Metric Report, ESSI Project 24,078 DTK GmbH, Germany, 1997

[5] E. V. Berard: Metrics for object-oriented Software Engineering, Object Agency, Internet document, 1998

[6] B. V. Binder: Testing Object-oriented Systems: A Status Report, Crosstalk Journal, Internet Document, 1995

[7] Harry Debler, S. Nordhoff: Process Metric Report, ESSI Project 24,078 DTK GmbH, Germany 1997

[8] Harry Debler: Boostrap Self-Assessment Report, ESSI Project 24,078 DTK GmbH, Germany 1998

[9] European Standard, CENELEC, prEN 50128, Railway applications – Software for railway control and protection systems, July 1998

[10] Garber / Heath Object-oriented Metrics, Initial Recommendation for C++ Systems, McCabe & Associates, 1997

[11] S. Humphrey, Software Configuration Management in: Managing the Software Process p113-246, Addison Wesley, Software Engineering Institute, 1989

[12] IEEE 828-1990; IEEE Standard for Software Configuration Management Plans, 1990

[13] IEEE 982.1-1988, IEEE Standard of Measures to Produce Reliable Software, 1988

[14] IEEE 982.2-1988, IEEE Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software, 1998

[15] IEEE 1061-1992, IEEE Standard for Software Quality Metrics Methodology, 1992

[16] A. Josub, Tool Selection Report, ESSI Project 24,078 DTK GmbH, Germany, 1997

[17] A. Josub, Project Programme, ESSI Project 24,078 DTK GmbH, Germany, 1997

[18] Kung D. C., Hsia P., Gao J.: Testing object-oriented Software, IEEE Coputer Society, 1998

[19] Latum / Solingen / Oivo / Hoisl / Rombach / Ruhe: Shifting to goal-oriented measurement in industrial environments, Schlumberger RPS, University of Kaiserslautern, 1996

[20] Lorenz/Kidd, Object-oriented Software Metrics, Prentice Hall, 1994

[21] B. Meyer: Object-oriented Software Construction, Second Edition, Prentice Hall, 1997

[22] Nordhoff, S: Enhanced development guidelines (EDG), ESSI Project 24,078 DTK GmbH, Germany, 1998

[23] Nordhoff, S: Enhanced quality manual (EQM), ESSI Project 24,078 DTK GmbH, Germany, 1998

[24] Nordhoff, S: Configuration Structure of the Baseline Project (CSK/CSS), ESSI Project 24,078 DTK GmbH, Germany, 1997/1998

[25] Nordhoff, S: Testing Process of the Baseline Project (TPK/TPS), ESSI Project 24,078 DTK GmbH, Germany, 1997/1998

[26] Bernd-Uwe Pagel: Software Engineering, Addison Wesley, 1994

[27] Ovum Evaluates: Configuration Management Tools, Ovum Ltd., UK, 1995

1998

[28] RATIONAL GmbH, ClearCase Concepts Manual, Document Number 4000-002-C, RATIONAL, 1998

[29] Rosenberg / Hyatt, Software Quality Metrics for object-oriented Environments, Goddard Space Flight Center , Crosstalk Journal, 1997

[30] Thomas C Royer: Software Testing Management, PTR Prentice Hall, 1993

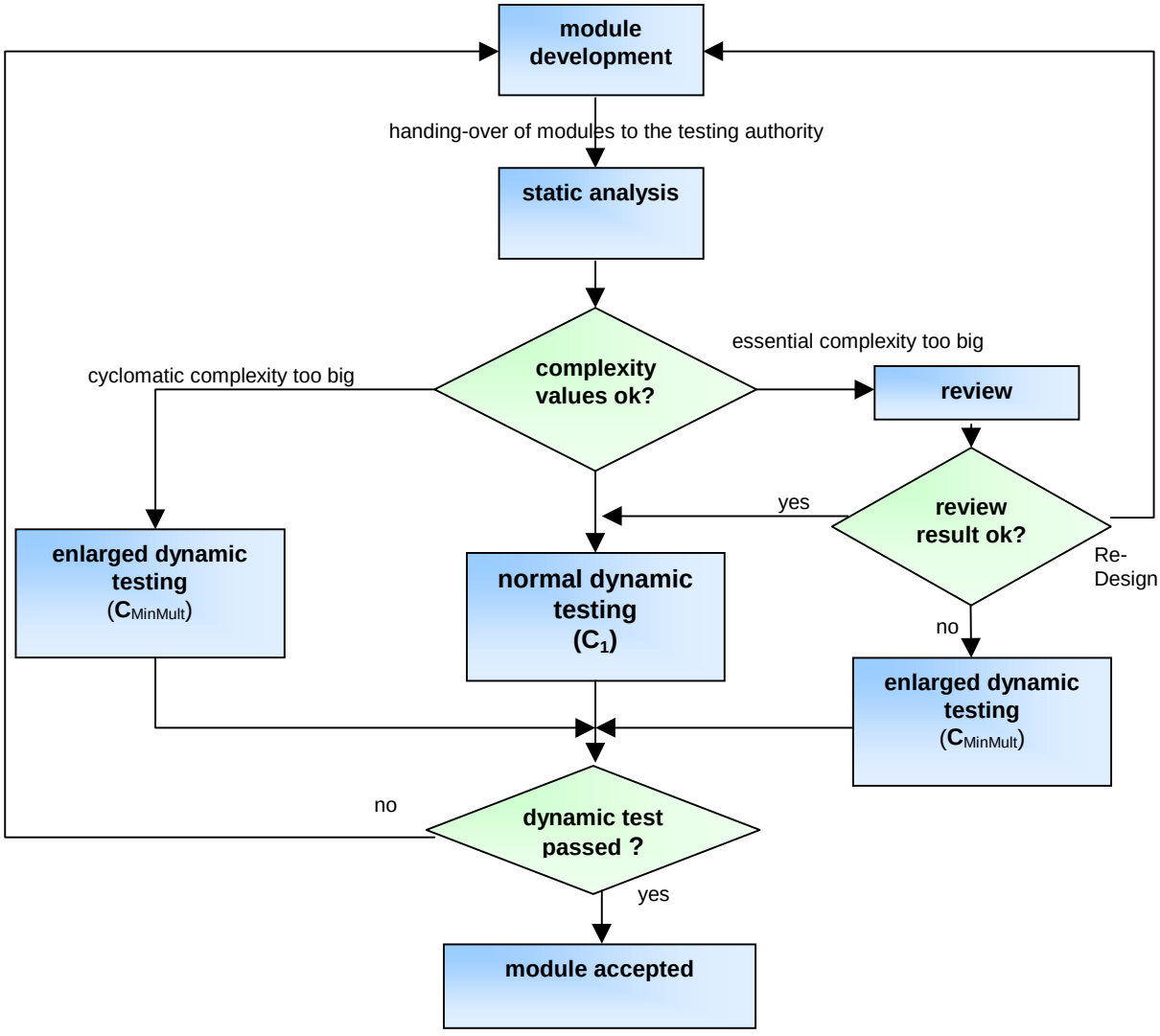
[31] Watson/McCabe, Structured Testing: A Testing Methodology using Cyclomatic Complexity, NIST Special Publication 500-2535, 1996

[32] Watt, H.D. Rombach: Practical benefits of goal-oriented measurement. In Proceedings of the Annual Workshop of the Centre for Software Reliability, pages 217-235. Elsevier, Sept. 1990

9 Annexes

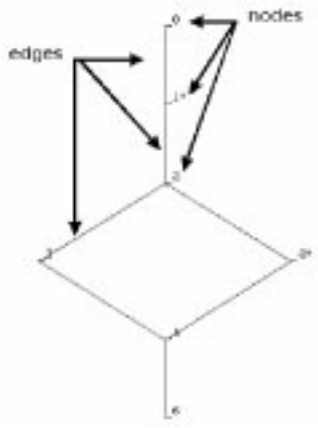
9.1 Test concept for module testing

The concept for module testing established during the PIE project is an example of static analysis procedures in co-operation with review activities and dynamic testing. Furthermore, the tracking and saving of testing states, attributes and data for this process is clearly necessary. The main aspect of the process control flow is the static analysis. In this stage the division into different complexity levels takes place. For this, we used the cyclomatic complexity measure to value the complexity structure and the essential complexity to value the structural development of the code. If the first value exceeds defined limits of our development guidelines [22] an enlarged dynamic testing procedure will be executed. If the essential complexity[2] of a tested function is to high, a review will take place. Depending on the result of the review a normal dynamic testing, an enlarged dynamic testing or a Re-design will be initiated.



The following metric description shows the advantages of the used metrics for module testing. In the PIE project additional metrics were defined (also object oriented metrics). For further explanation see the process metric report [4].

Cyclomatic complexity



Cyclomatic complexity is a measure of the complexity of a module's decision structure. It is the number of linearly independent paths, and, therefore, the minimum number of paths that should be tested to reasonably guard against errors. A high cyclomatic complexity indicates that the code may be of low quality and difficult to test and maintain. In addition, empirical studies have established a correlation between high cyclomatic complexity and error-prone software [31].

Advantages are:

- Identifies error-prone software.
- Pinpoints overly complex modules that may require subdivision.
- Measures the minimum testing effort and reveals areas of concentration for testing.

$$V(G) = e - n + 2$$

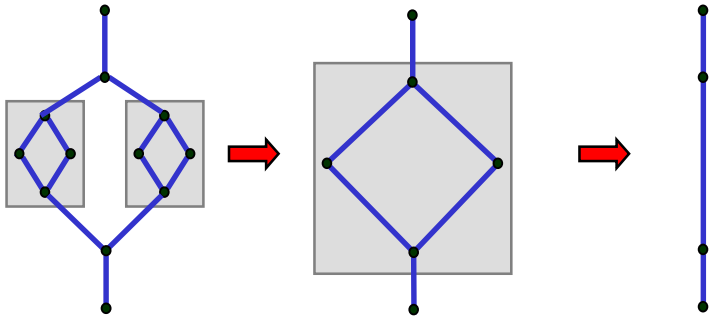
with e = number of edges in the control flowgraph

with n = number of nodes

Essential Complexity

Essential complexity is a measure of the degree to which a module contains unstructured constructs. Unstructured constructs decrease the quality of the code and increase the effort required to maintain the code and break it into separate modules. Therefore, when essential complexity is high, there is a high number of unstructured constructs, so modularization and maintenance is difficult. In fact, during maintenance, fixing a bug in one section often introduces an error elsewhere in the code [31].

Essential complexity is calculated by removing all structured constructs from a module's flowgraph and then measuring the cyclomatic complexity of the reduced flowgraphs.



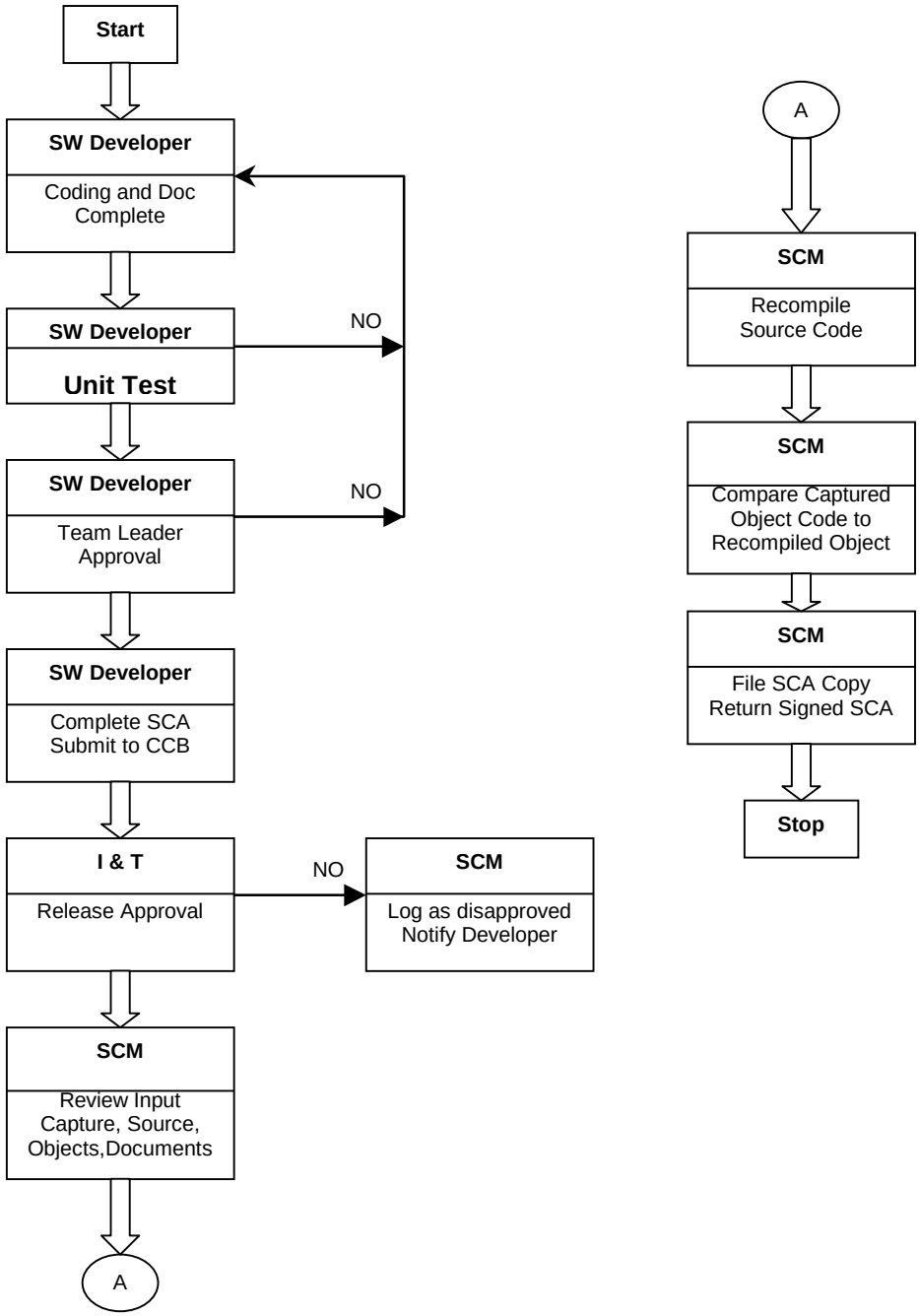
Advantages are:

- Measures the degree of unstructuredness.
- Reveals the quality of the code.
- Predicts the effort required to maintain the code and break it into separate modules.

[2] This measure should be used carefully, since the calculation points not necessarily under each condition to unstructured code.

9.2 Process flowgraph of the configuration management process

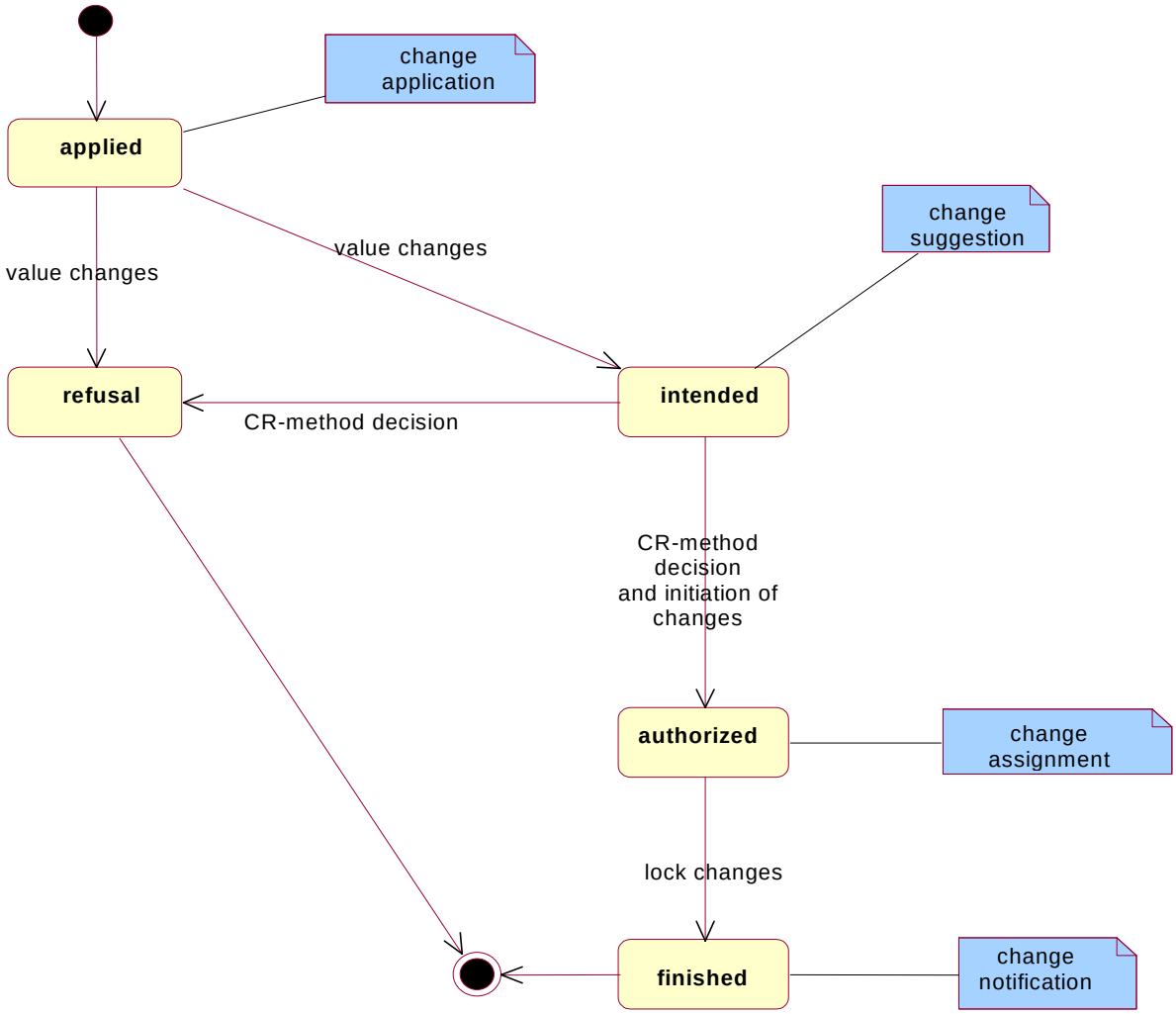
The process flowgraph for the configuration management



9.3 Process state diagram of the change request process

This state diagram shows the different states of a change request inquiry, solving a product or software demand[3], produced within the software development process [1]. For a general introduction, this process is concerned to all objects produced in the development life cycle (e.g. analysis documents like use cases, design documents like class diagrams, implementation documents like source files).

CR-demands cover different steps of processing, represented in the model with the usage of states. A CR-object in a specific state enabling only a well defined responsible crew to work on this object. For a change application (=demand is in state applied or intended) the testing staff is excluded. For this, a CR-co-ordinator and a CR-committee must be introduced deciding about a CR demand. Only if a positive decision is produced (state is authorised) the development has the responsibility to solve the CR demand.



[3] The word “demand” is defined in the meaning of changes and extensions, faults or errors [23] .

9.4 Configuration management responsibilities

The definition of responsibilities in the configuration management process is an important task. For this, we established for each part of the baseline project (kernel and subset) configuration management structures identifying, amongst other things, several aspects of responsibilities.

Responsibilities	Program Manager	Software Engineer	CMA Authority	SQA
Configuration identification	Review	Originate	Originate	
Baseline definition	Review	Originate	Originate	Approve
Change preparation	Review	Originate	Review	Approve
Change control	Review	Review	Originate	Approve
Change implementation	Review	Originate	Review	Approve
Status accounting	Review	Review	Originate	Approve
Formal SCM audits	Review	Review	Originate	Approve

Table 2 Responsibility Assignments

9.5 Version and release management

The introduction of a version and release management system provides the possibility to track and store different version of a source file. The ability to track down all changes and extensions to sources and to store this changes into versions extended the reliability and maintainability of the development process. For further implementation, we established configuration baselines for the different development stages based on the V-Model [1].

This is a sample version tree for a development baseline. The *REL1.0a* label identifies a version with status of alpha-checked sources. This sources are handed to the testing staff. If errors are detected a branch for bug-fixing will be generated storing the fixed sources. The next step is the merging action of the bug-fixed sources into the main development branch. This action is important if different development staff members are working on different releases of a product (e.g. development and bug-fix release)

Figure 1 Version tree with releases and merging operation

The next figures shows the usage and graphical representation of the configuration management system. The developer could use an Explorer-based file-manager implementing the important information and version numbers.

Figure 2 Configuration management explorer

Furthermore, the developer has the ability to compare different versions of a source file to identify changes or extensions.[4] This function is very helpful to track crept bugs in the different version of a source file. As a extension the configuration management system has the ability to compare also Microsoft WORD® files.

Figure 3 Comparing of different versions of files

[4] This functionality is only usable on text files

9.6 Test Tools and usage

The next table lists the different test tools that have been introduced during the PIE project. We split the different tools into the main areas of usage to provide a better understanding of the main areas covered by the tools.

test area	*Tool*	*Supplier*
Static analysis - *analysis of code with metrics* - *analysis with flowgraphs* - *analysis of internal data structures* - *analysis of quality aspects (metric driven)* - *OO support* - *graphical support (e.g. flowgraphs)* - *supports different programming languages*	McCabe Visual Quality ToolSet	McCabe & Associates
Dynamic testing - *black-box testing* - *white-box testing* - *class testing* - *stubs and driver approach*	Cantata 3 / Cantata++	IPL
Capture Replay - *automated testing* - *capturing and scripting of GUI functionality* - *simulating user actions* - *embedded within projects of Visual Studio*	Visual Test	RATIONAL
Memory Leak detection - *detects uninitialized memory reads* - *detects memory allocation errors* - *detects memory leaks* - *detects array bound errors* - *detects accesses through dangling pointers* - *instrumenting programs* - *integration in Visual Studio*	Purify NT	RATIONAL

9.7 BOOTSTRAP evaluation results

The baseline project has undergone a BOOTSTRAP-compliant Self-Assessment in 1995. This Assessment intended to serve as the basis for a valuation of the defined process methods and procedures. Through the help of a second Self-Assessment in June 1998, at the end of the ESSI-PIE project, the possibility to value the results was given. These results determined which significant improvements have been reached [8].

The Self-Assessment was performed with support of the tool BootCheck 3.0 which is based on the assessment method BOOTSTRAP 3.0.

The complete BOOTSTRAP 3.0 method calculates maturity levels between 0 and 5 for the processes in question. Whereas, for BootCheck 3.0 the maximum evaluated maturity level is 3.

The **maturity level 1** explains in common that the process in question is fulfilling the defined purpose. Base practices within the process are generally performed, however, they are not precisely planned nor pursued. The execution is dependent on individual efforts and knowledge. There exists recognisable work products for the process. If one process shows **maturity level 2** it means that the processes' execution is already planned and pursued. The process delivers work products of an acceptable quality and this within a defined time frame. Products are conform with defined standards and requirements. The execution of the observed process at **maturity level 3** is standardised over and above the project, tailored towards the needs, and precisely documented. It is guaranteed that all necessary recourses for a process execution are available.

The assessment from 1995 revealed maturity levels between 1 and 1.5 within the area of test relevant processes as well as for Configuration Management.

The assessment performed after the improvement of the PIE have been taken place, shows a maturity level of 2.5 for Configuration Management and for the processes „Software Implementation and Testing“ and „Software Integration and Testing“. This means a significantly increase of the maturity of the processes which were the focus of the PIE.

Especially the test activities introduced in the course of the PIE have lead to revaluation. As the quality of the performed verifications, before the improvement activities have taken place, was very much depending on peoples knowledge, now all test sub processes are much more institutionalised. Systematic procedures have been defined for Module and Integration Testing, which includes suitable test methods and secure re-testability. With this, it is also warranted, that an adequately high testing coverage of the functional and structural complexity of the system will be reached. A test regression strategy has been established on different test levels (Module Test, Integration Test). This guarantees, after software modifications, that the used product quality is being kept and that the possibility of unwanted side effects is being reduced. In order to support introduced testing methods, suitable test tools have been implemented which increases the effectiveness of the test actions. With regard to the introduced object oriented techniques special test methods have been introduced on this subject, which fits the technology's requirements. Examples therefore are Class Tests, Cluster Tests, and Model Tests. All documents, which stand in relation to test activities, are subjugated to the Configuration Management Process. For the future it is recommendable to develop an approach for the mandatory conduction of reviews/inspections for the results of the performed tests.

„System Integration and Testing“, which was not focus of the PIE is a Life-Cycle process which needs further improvement in the future. System Testing should be developed to a similar high maturity where Module Testing and Integration Testing already are. Therefore it should be

taken into account to define subsystems for testing instead of testing only the complete system. The enhancement of the existing guidelines and procedures for System Testing which are already in progress should be pushed.

The CM installed contains the administration for the entire of the objects generated in different project phases in the course of the developing process. The CM process, which is precisely adapted to the project requirements, is supported through the insertion of the CM tool ClearCase. All types of documents are administrated under CM. The mechanisms, roles, guidelines, etc. are described in a CM plan. The experiences have been disseminated to other projects in the company. One of CM's areas for further improvement is the 'Change Request Management'. The goal oriented seizing, classifying, prioritising, and performing of change requests should be expanded. Herewith, it is also to be thought of a tracking of errors concerning individual project phases. Concerning the choice of tools for the support of an integrated 'Change Request Procedure', it is important, amongst other things, to consider a perfect integration into the existing tool environment.

9.8 Quality characteristics

The McCabe Visual Toolset, which we used for static analysis purposes provides several metrics. One type of graphical representation of a static analysis is the **Kiviat diagram**. In the Kiviat diagram each metric is represented as a ray with the absolute minimum in the centre of the diagram. The threshold of each metric is plotted at the centre of its ray. The average and the maximum values of the metrics are plotted in proportion to the threshold and minimum. For the representation we chose typical values for complexities as threshold values. For the cyclomatic complexity (v(G)) the defined threshold is *10*, for the essential complexity (ev(G)) *4*, for the Module Design Complexity (iv(G)) *7*, for number of lines (nl) *60* and for the Normalised cyclomatic complexity *0.3*.

The interrelationships of these metrics can be interpreted as quality aspects e.g. 'Maintainability', 'Testability'.

(a) (b)

Figure 4 Quality aspects for the baseline kernel (a) and the baseline subset (b)

Figure 5 shows a comparison between the baseline kernel and the baseline subset using Kiviat diagrams. In the baseline subset the maximum values for the metrics have decreased.

9.9 Distribution of complexity

The **scatterplot** shows the relationships of essential and cyclomatic complexity.

Each module is represented by an asterisk and is plotted in relation to the threshold values of v(G) and ev(G). The modules asterisks in the lower left quadrant indicate low cyclomatic and low essential complexities, therefore these modules are classified as reliable and maintainable. The modules in the lower right quadrant have a low essential complexity and a high cyclomatic complexity, therefor these are maintainable, but less reliable. The upper left quadrant is containing modules which are reliable, but harder to maintain. The upper right quadrant shows the most critical modules, due to the cyclomatic and essential complexity. This is the area of unreliability and unmaintainability. The modules in the upper right quadrant are the first candidates for more intensive test activities.

As one can see easily, the majority of the modules in the baseline kernel as well as in the baseline subset is located in the lower left quadrant of the scatterplot. Nevertheless in the baseline kernel there are several functions in the area of 'Unreliable/Unmaintainable' of the scatterplot, which are categorised as high critical from the McCabe metrics. In the baseline subset a substantial improvement concerning maintenance and reliability has occurred.
unreliable and unmaintainable code.

(a) (b)

Figure 5 Distribution of complexity for the baseline kernel (a) and the baseline subset (b)

9.10 Object oriented Aspects of quality

Another strong point in static analysis was the evaluation of object oriented metrics. There is very less fundamental knowledge available on this topic, especially no experience based knowledge. Therefore we work more or less intuitively. As the focus in the functional metrics is on software complexity, the main point of the object oriented metrics is more on the program code and the program structure, than on complexity aspects. This fact makes it more complicate to select the most appropriate metrics for the concerned application. Table 3 is a summary of the metrics we selected together with the quality characteristics they are influencing.

(a) (b)

Figure 6 OO quality aspects for the baseline kernel (a) and the baseline subset (b)

The Kiviat diagrams in Figure 6 are a presentation of the values measured by the McCabe Visual Toolset for the object oriented metrics of the analysis of the baseline kernel and subset. The extensive usage of protected data leads to high '%Pub' metric values. The usage of protected attributes for inheritance is not necessarily a problem, but should be used carefully.

9.11 Valuation of metrics

This table shows the main effects and valuation areas of metrics used within our project. To cover this statement with scientific results we used footnotes to identify the relation to authors and their documents. We restrict our evaluation to product metrics, since these metrics gave us a better understanding estimating our product in the meaning of complexity, reliability and testability.

quality characteristics / metric	Complexity	Reliability	Maintainability	Testability	Understandability	Reusability	Efficiency	more error prone
Module-Metrics								
v(G)[5]	X		X	X				X
ev(G)[5]			X	X				X
iv(G)[5]			X	X		X	X	
S0[5]	X		X	X			X	
S1[5]				X				
nl[5]	X							
nv(G)[5]	X	X						
GDV(g)[5]	X		X	X			X	
OO-Metrics								
Object S1				X				
WMC[6, 7, 8]	X		X		X	X		X
DIT[6, 7, 8]				X	X			X
NOC[6, 7, 8]				X		X	X	
CBO[6, 7, 8]				X		X	X	X
RFC[6, 7, 8]	X		X	X	X			
LCOM[6, 7, 8]						X	X	X
sum_vg	X							
FanIn		X	X	X	X			
PCTCALL[8]						X	X	
PCTPUB[8]			X	X	X			
PUBDATA[8]			X	X		X		
NAC[8]			X		X	X	X	

Table 3 Valuation of metrics

[5] see Watson/McCabe [31]
[6] see Rosenberg / Hyatt [29]
[7] see Basili / Briand / Melo [3]
[8] see Garber / Heath 10]

9.12 Additional numeric results

9.12.1 Degree of documentation

The percentage of comments of the biggest functions of the analysed software is presented in Figure 7. In our company wide development guidelines it is recommended to have a degree of documentation of 20%. This requirement is satisfied by the majority of the functions.

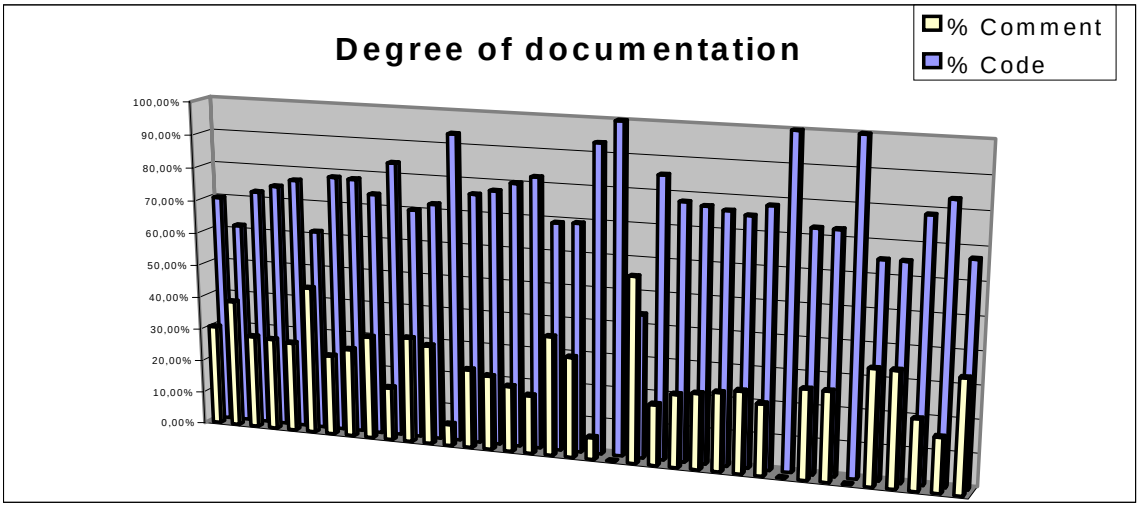


Figure 7 Degree of documentation

9.12.2 Complexity / Errors

Figure 8 shows the distribution of errors in correlation to the McCabe complexity and the number of test cases generated. As a matter of fact most of the errors are detected in modules with a complexity higher then 10.

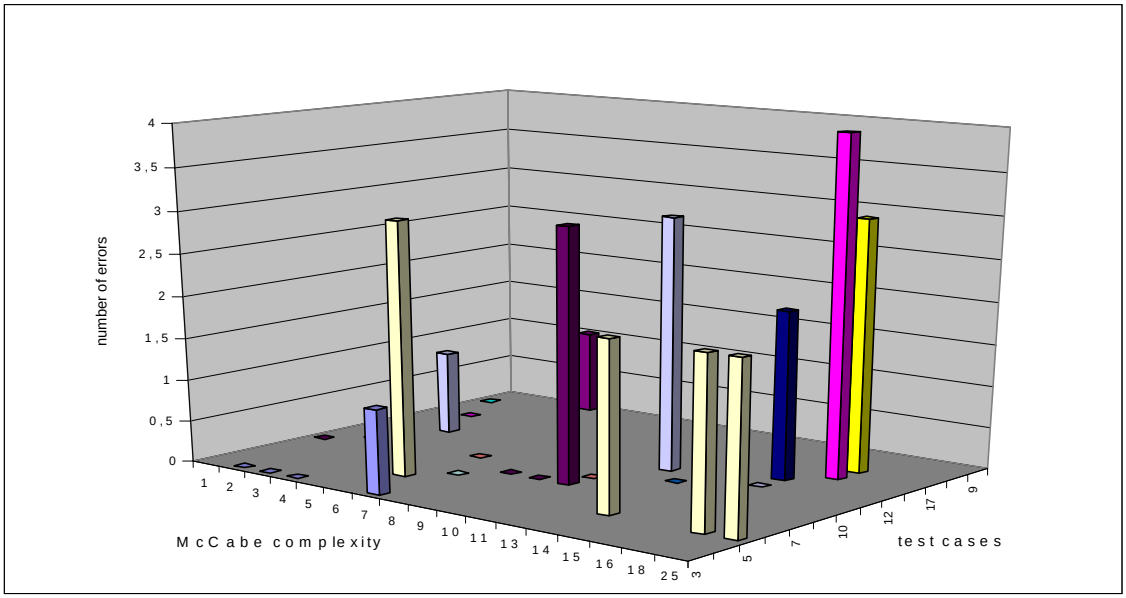


Figure 8 Complexity / Errors

9.12.3 Error distribution in the test phases

Figure 9 is a presentation of the distribution of errors to different test phases, respectively life cycle phases. Most errors are detected in early phases (module and integration test). During the system test we found about 25% of the total amount of errors and most of them were GUI errors.

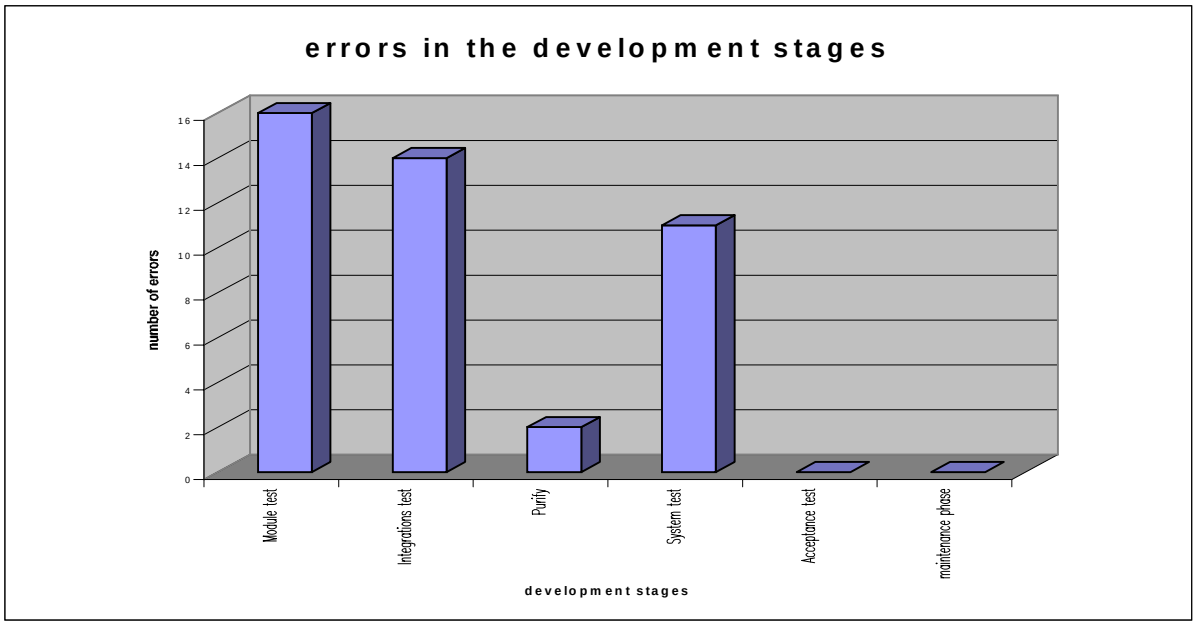
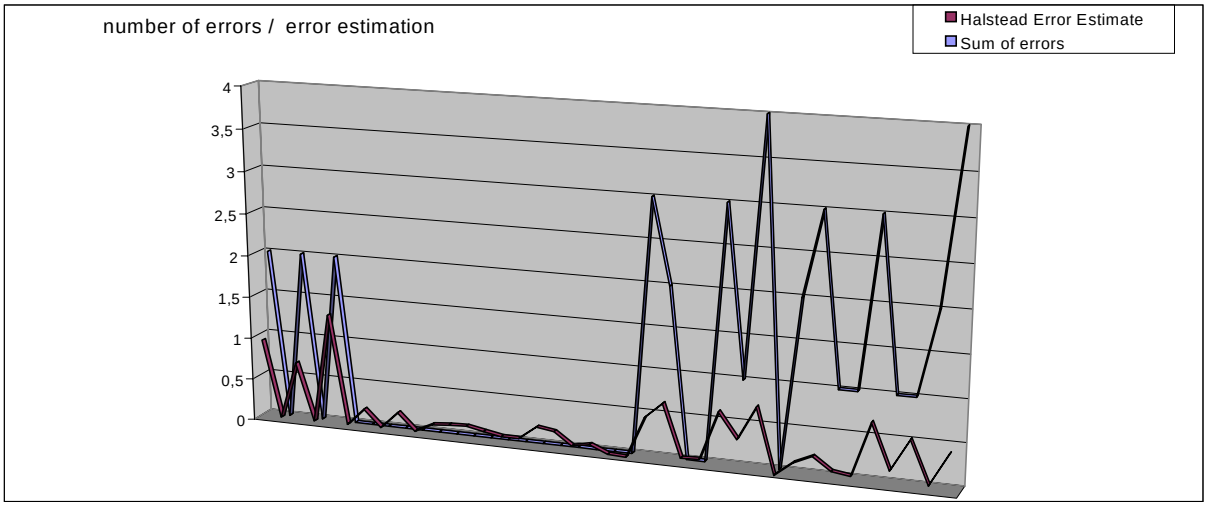


Figure 9 Errors in the test phase

9.12.4 Error estimation

In the set of metrics created by Halstead there is one specific which is designed for error estimation. We used this metric for the analysed functions to compare the forecasted error values with the real number of errors we detected in the concerned functions during the performed test actions. Although there is no statistical expressiveness due to relatively small number of data, it is interesting to see the relation between predicted and determined



values.

Figure 10 Error estimation